2008

Computationally efficient resource allocation for complex system reliability studies

Jessica Chapman
Iowa State University

Computationally efficient resource allocation for complex system reliability studies

by

Jessica Lynn Chapman

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Statistics

Program of Study Committee:
Max D. Morris, Major Professor
Michael Larsen
William Q. Meeker
W. Robert Stephenson
Stephen Vardeman
Christine Anderson-Cook

Iowa State University

Ames, Iowa

2008

DEDICATION

I would like to dedicate this thesis to my parents, Markus and Terry Chapman.

TABLE OF CONTENTS

LIST OF TABLES vi

LIST OF FIGURES x

ACKNOWLEDGEMENTS xiii

CHAPTER 1. INTRODUCTION 1

1.1 Introduction to Complex System Reliability 2

1.2 A Hierarchical Model for Combining Multiple Information Sources to Assess System Reliability 5

1.3 Implementation 8

1.3.1 System Representation 9

1.3.2 Random-Walk Metropolis-Hastings 11

1.4 Introduction to Resource Allocation 14

CHAPTER 2. EVALUATION OF CANDIDATE ALLOCATIONS 16

2.1 Expected Information Gain 17

2.2 Current Approach for Comparing Candidate Allocations 20

2.3 Proposed Approach for Comparing Candidate Allocations 21

2.4 Implementation of Proposed Approach 27

2.4.1 Example: Evaluating Candidates for a Five Component Series System . 28

2.5 Numerical Error in the Histogram-Based Calculation of Entropy for Beta Distributions 31

2.6 Evaluation of Large Candidate Allocations 38

2.6.1 Demonstration: Evaluating Large Candidate Evaluations Efficiently . . 41

2.7 Chapter Summary 43

CHAPTER 3. GENETIC ALGORITHMS FOR OPTIMAL RESOURCE ALLOCATION STUDIES 46

3.1 Introduction to Genetic Algorithms 47

3.1.1 Selection 48

3.1.2 Reproduction 50

3.2 Implementation of a Genetic Algorithm 52

3.2.1 Generating the Initial Population 52

3.2.2 Parent Selection and Recombination 53

3.2.3 Mutation 54

3.3 Example: Using Genetic Algorithms to Find an Optimal Resource Allocation . 56

3.4 Genetic Algorithms for Large Systems and Allocations 60

3.4.1 Example: GA for Eight Component Series/Parallel System with a Single Expert 61

3.4.2 Example: GA for Eight Component Series/Parallel System with Two Experts 69

CHAPTER 4. CASE STUDIES 73

4.1 Air-to-Air Heat-Seeking Missile 73

4.2 Demand Unavailability of the Low-Pressure Coolant Injection System in Nuclear-Power Boiling-Water Reactors 91

CHAPTER 5. SUMMARY AND DISCUSSION 103

5.1 New Methodology for Candidate Evaluation (Chapter 2) 103

5.2 Genetic Algorithms for Resource Allocation (Chapter 3) 105

APPENDIX A. OUTLINE OF GENETIC ALGORITHM FOR RESOURCE ALLOCATION 107

APPENDIX B. ADDITIONAL MATERIAL FOR MISSILE CASE STUDY 110

APPENDIX C. ADDITIONAL MATERIAL FOR LPCI SYSTEM CASE STUDY 126

BIBLIOGRAPHY 133

LIST OF TABLES

Table 2.1 Entropy for Beta distributions with $\alpha = 10$, β=5, 10, 20, 30, 40, and 50 20

Table 2.2 Joint first-stage posterior draws from an MCMC sampler 22

Table 2.3 System reliability posterior draws and computed outcome probabilities for a candidate allocation $\mathbf{n}_2$ 23

Table 2.4 First-stage data, expert best guesses, and tests costs for five component series system 29

Table 2.5 Expected entropy for the five candidate allocations 30

Table 2.6 Numerical error in the histogram-based entropy calculation using bin width h for various Beta distributions 35

Table 2.7 Expected entropy, computed using 100 bins, for the five candidate allocations and approximate error bands 37

Table 2.8 Expected entropy, computed using 721 bins, for the five candidate allocations and approximate error bands 37

Table 2.9 Effect of increasing system size and experiment size on the number of possible outcomes 40

Table 2.10 The expected entropy compute for each candidate allocation and the estimate of expected entropy, standard deviation of the sampled entropies, and 95% confidence interval for the expected entropy, based on samples of size N=100, 500, 1000, 5000, and 10,000 45

Table 3.1 First-stage data, expert best guesses, and component testing costs for eight component series system 56

Table 3.2 First-stage data, expert guesses, and testing costs for simple eight component series/parallel system (Hamada *et al* (2008)) 62

Table 3.3 Estimated expected entropy and standard deviation for candidates in the final population of genetic algorithm as evaluated by the genetic algorithm, using samples of size 1000. These 40 allocations, as well as the two allocations described by Hamada *et al* (2008) were re-evaluated using samples of size 10,000 68

Table 3.4 Expert best guesses from two experts 69

Table 3.5 The 63 unique candidate allocations in the final populations of two runs of the GA were re-evaluated using samples of size 10,000 and the estimate of the expected entropy, the standard deviation, and a 95% confidence interval for the expected entropy of the allocation were computed 72

Table 4.1 First-stage data and expert best guesses for series missile system (Martz *et al* (1988)) 75

Table 4.2 Hypothetical testing costs for testable components in missile system . 78

Table 4.3 Candidate allocations in the final population of the first run of the GA 80

Table 4.4 Expected entropy for unique candidate allocations from the final populations of two runs of the GA 82

Table 4.5 Second hypothetical testing costs for testable components in series missile system 83

Table 4.6 Best allocation found by GA for second cost structure 84

Table 4.7 Final population in best GA for second cost structure for the missile case study 85

Table 4.8 Final population in best GA for first cost structure for the missile case study with a budget of \$10,000; the allocations are represented as the number of tests of each of the following components: (C_3, C_5, C_{11}, C_{15}, C_{16}, C_{17}, C_{18}, C_{19}, C_{20}, C_{21}, C_{22}, C_{23}, C_{24}, C_{35}) 87

Table 4.9 Final population in best GA for second cost structure for the missile case study with a budget of \$10,000; allocations represent the number of tests on the following components: (C_3, C_9, C_{11}, C_{13}, C_{16}, C_{17}, C_{18}, C_{19}, C_{20}, C_{21}, C_{22}, C_{23}, C_{24}, C_{35}) 89

Table 4.10 First-stage data and expert best guesses for LPCI system (Martz and Waller (1990)) 93

Table 4.11 Best allocation in each of the three runs of GA using $h \approx \frac{\hat{\sigma}}{6}$ 96

Table 4.12 Estimate of the expected entropy of the unique candidate allocations from the final populations of the three runs of GA using $h \approx \frac{\hat{\sigma}}{6}$ and sample size N=10,000 to evaluate allocations 97

Table 4.13 Candidate allocations that use the available resources to collect more tests on the basic components that had first-stage failures 98

Table 4.14 Best allocation in each of the three runs of GA using $h \approx \frac{\hat{\sigma}}{10}$ 99

Table 4.15 Best allocation in each of the three runs of GA using $h \approx \frac{\hat{\sigma}}{16}$ 99

Table 4.16 Best allocations found by the three GA runs using bin widths $h_1 \approx \frac{\hat{\sigma}}{6}$, $h_2 \approx \frac{\hat{\sigma}}{10}$, and $h_3 \approx \frac{\hat{\sigma}}{16}$, each re-evaluated using samples of size $N = 10,000$ and all three bin widths 100

Table B.1 Major subsystems and basic components in air-to-air heat-seeking missile system (Table 1 from Martz *et al* (1988)) 111

Table B.2 Induced, Native, and Combined Prior Distributions of Martz *et al* (1988) and Posterior Distributions for Major Subsystems and Missile System . 113

Table B.3 Estimated posterior means, standard deviations, and quantiles, with naive and batch means Monte Carlo SE estimates from the R package coda, for the 41 model parameters 121

Table B.4 Sensitivity of system reliability posterior distribution to choice of prior distribution for the parameters N, J, and γ 124

Table C.1 Beta posterior distributions and posterior quantiles for subsystem and LPCI system demand availability 127

Table C.2 Estimated posterior means and standard deviations, with naive and batch means Monte Carlo SE estimates, for the 26 model parameters . 130

Table C.3 Estimated posterior quantiles for the 26 model parameters 131

Table C.4 Sensitivity of system reliability posterior distribution to choice of prior distributions for N, J, and γ 132

LIST OF FIGURES

Figure 1.1 Reliability block diagrams for series and parallel systems 2

Figure 1.2 Event trees for series and parallel systems 3

Figure 1.3 Event tree for missile system (Johnson *et al* (2003)) 4

Figure 1.4 Event tree for five component series system 10

Figure 2.1 Probability density functions for Beta distributions with $\alpha = 10$, β=5, 10, 20, 30, 40, and 50 19

Figure 2.2 System reliability prior (dashed) and posterior (solid) distributions using the model described by Johnson *et al* (2003) 30

Figure 2.3 Numerical error in the histogram-based entropy calculation for Beta distributions with specified mean and c_v 33

Figure 2.4 Numerial error in the histogram-based entropy calculation for bin widths (a) $h = \frac{\sigma}{2}$, (b) $h = \frac{\sigma}{4}$, (c) $h = \frac{\sigma}{6}$, and (d) $h = \frac{\sigma}{10}$ 34

Figure 3.1 Event tree for eight component series system 57

Figure 3.2 Posterior distribution of system reliability given first stage data 58

Figure 3.3 Best-so-far curve displays the expected entropy of the best allocation at each generation 58

Figure 3.4 Tests per Component for the Best Allocation at Each Generation . . . 59

Figure 3.5 Simple eight component series/parallel system (Figure 9.2 from Hamada *et al* (2008)) 61

Figure 3.6 Prior (dashed) and posterior (solid) distributions for system reliability (Figure 9.3 from Hamada *et al* (2008)) 63

Figure 3.7 Tests per components at each generation (Figure 9.8 from Hamada *et al* (2008)) 64

Figure 3.8 Prior (dashed) and posterior (solid) distributions of system reliability under model from Section 1.2 65

Figure 3.9 Performance of a single run of a genetic algorithm to find an optimal resource allocation for the simple eight component system of Hamada *et al* (2008) 66

Figure 3.10 Composition of the best allocations found in two runs of a genetic algorithm to find an optimal resource allocation for the simple eight component system when best guesses were specified by two experts 70

Figure 4.1 Event tree for series missile system 74

Figure 4.2 Prior distributions for the reliability of the five subsystems and missile system (solid = Johnson model, dashed = combined prior distributions of Martz *et al* (1988)) 76

Figure 4.3 Posterior distributions for the reliability of the five subsystems and system (solid = Johnson model, dashed = Martz *et al* (1988)) 77

Figure 4.4 Performance of a single run of the genetic algorithm for finding an optimal allocation of resources in the missile case study under the first cost structure 79

Figure 4.5 Performance of a second run of the genetic algorithm for finding an optimal allocation of resources in the missile case study under the first cost structure 81

Figure 4.6 Performance of a single run of the genetic algorithm for finding an optimal allocation of resources under second cost structure in the missile case study 84

Figure 4.7 LPCI system demand availability block diagram (Figure 1 from Martz and Waller (1990)) 91

Figure 4.8 Prior distributions for the reliability of subsystems and LPCI system from Johnson *et al* (2003) model (solid) and Martz and Waller (1990) model (dashed) 94

Figure 4.9 Posterior distributions for the reliability of subsystems and LPCI system of Johnson *et al* (2003) model (solid) and Martz and Waller (1990) model (dashed) 94

Figure B.1 Autocorrelation plots for Basic Components 1-16 117

Figure B.2 Autocorrelation plots for Basic Components 17-32 117

Figure B.3 Autocorrelation plots for Subsystems, N, γ, and J 118

Figure B.4 Density estimates for Basic Components 1 - 16 119

Figure B.5 Density estimates for Basic Components 17 - 32 119

Figure B.6 Density estimates for Subsystems, System, N, γ, and J 120

Figure B.7 Prior distributions for N and J in the sensitivity analysis 123

Figure C.1 Autocorrelation plots for Components 1 - 16 128

Figure C.2 Autocorrelation plots for Components 17 - 23, N, γ, and J 129

ACKNOWLEDGEMENTS

I would like to thank my adviser, Dr. Max Morris, for all of his guidance and support in the research and writing of this dissertation. I would also like to thank Dr. Christine Anderson-Cook of Los Alamos National Laboratory and Dr. Alyson Wilson of Iowa State University (formerly of Los Alamos National Laboratory) for introducing me to this problem and all of their encouragement of this work. This work was supported in part by NSF grant DMS #0502347 EMSW21-RTG awarded to the Department of Statistics, Iowa State University.

CHAPTER 1. INTRODUCTION

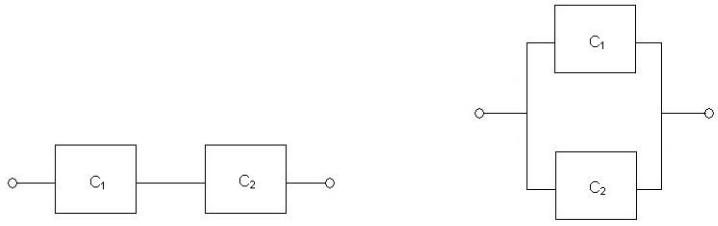
Data collection planning (Hamada *et al* (2008), Chapter 9) is an important step in the experimental design process. The context in which we address data collection planning is that of the second-stage in complex system reliability studies; this second-stage data will be used to obtain a more precise estimate of the system's reliability. We assume that first-stage data from previous studies are available at various system levels and that, through a Bayesian analysis, estimates of the reliability of the system and the system components are available. Using the information from the initial analysis, we would like to plan how to collect second-stage data. This problem is often referred to as "resource allocation" (Hamada *et al* (2004), Wilson *et al* (2006)).

In this chapter, we build the framework necessary to discuss resource allocation for complex system reliability studies by first giving a brief introduction to the field of complex system reliability. One of the primary goals in system reliability studies is to assess the reliability of the full system. Martz *et al* (1988), Martz and Waller (1990), and Johnson *et al* (2003) have addressed the problem of incorporating both data from various levels of the system and expert guesses about the reliability of system components to estimate system reliability. We summarize the work of Johnson *et al* (2003), which will serve as our framework for addressing resource allocation for complex system reliability studies, and describe our implementation of their model. We also describe the motivating problem of resource allocation for complex system reliability studies and give an overview of the dissertation.

1.1 Introduction to Complex System Reliability

A complex system is a system composed of many components or subsystems that must work together in order for the full system to work successfully. Examples of complex systems can be found in many different scientific and engineering disciplines and are as varied as missile systems, power plants, cars and the human body. We assume that the system is comprised of subsystems, which may be further decomposed into basic components; basic components represent the lowest level of the system and can't be decomposed further. The definition of labels assigned to various parts in the system is partially subjective, is related to the level of granularity to which we wish to decompose the system, and is dependent on what data are available or can be acquired. The basic components, subsystems, and system will be referred to generically as *components*.

Components in the complex system can be connected either in series or parallel; more complicated system structures will involve both series and parallel connections. In series systems all components must work in order for the system to work, while in parallel systems at least one component must work in order for the system to work. Reliability block diagrams for two-component series and parallel systems are shown in Figures 1.1(a) and 1.1(b), respectively. In this work, we are assuming that the reliabilities of the components and the full system do not change over time.

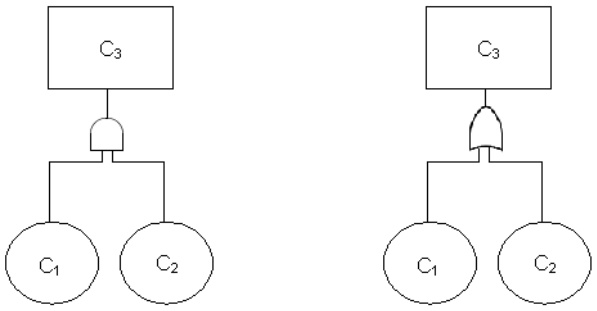


(a) Series System (b) Parallel System

Figure 1.1 Reliability block diagrams for series and parallel systems

For a complex system with k components, we will denote the reliabilities of the components as p_1, p_2, ..., p_k, with the highest index representing the full system. It is assumed that the connections in the system are perfect (i.e., a system will fail only if one or more of its

components fail). Anderson-Cook (2009) describes how this assumption can be tested. In a series system the reliability of the full system is the product of the basic component reliabilities. An event tree for a two-component series system is displayed in Figure 1.2(a); for this sample system, the system reliability is the product of the reliability of the two components, $p_3 = p_1 p_2$. In a parallel system the system reliability is 1 minus the probability that all basic components fail. Figure 1.2(b) displays an event tree for a two-component parallel system. For this sample system, the system reliability is $p_3 = 1 - (1 - p_1)(1 - p_2)$.



(a) Series System (b) Parallel System

Figure 1.2 Event trees for series and parallel systems

A primary goal in complex system reliability is to estimate the reliability of the full system. Johnson *et al* (2003) and Anderson-Cook *et al* (2007) discuss system reliability in the context of weapons stockpile reliability. Some of the units in these stockpiles, such as missile systems, are composed of smaller components and subsystems. The event tree for such a missile system is displayed in Figure 1.3. Due to degradation over the storage period, these units may fail to perform their intended functions. The interest is in estimating the probability that a randomly selected unit from the stockpile will function properly (Anderson-Cook *et al* (2007)). Martz *et al* (1988) describe the problem of estimating the reliability of an air-to-air heat-seeking missile, with five major subsystems, under specific use conditions. Martz and Waller (1990) address demand unavailability for a 1,150 megawatt electric U.S. commerical nuclear-power boiling-water reactor. Wilson *et al* (2006) give other examples of complex systems, including nuclear weapons, infrastructure networks, and supercomputer codes.

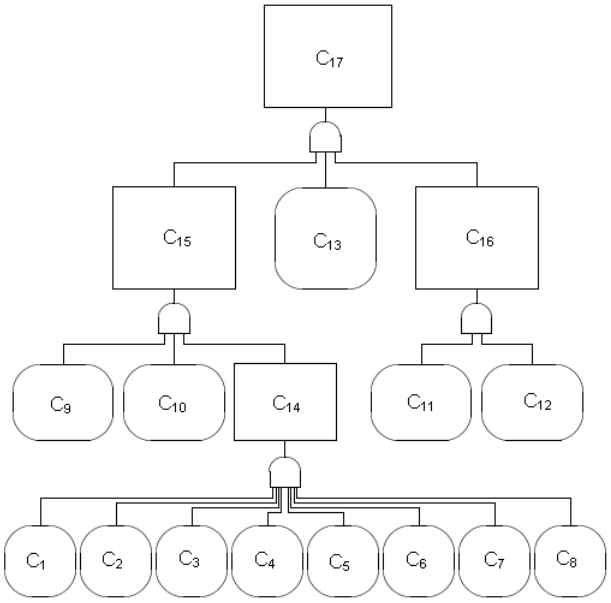


Figure 1.3 Event tree for missile system (Johnson *et al* (2003))

The most direct way to estimate system reliability would be to perform full system tests. However, in many complex systems the availability of full system data may be limited due to the cost associated with system tests. There are, however, other sources of information that can indirectly tell us about the reliability of the system. Quality assurance data, maintenance data, and measurements from common components in similar systems are examples of other potential sources of data. In addition, information in the form of system expert opinion may be available at the basic component, subsystem, and/or system level. With data available at different levels of the system, and different types of information available, the challenge becomes how to combine this information to learn about the reliability of the system.

Martz *et al* (1988) and Martz and Waller (1990) were the first to provide a means for which both data and expert opinion, available at any level of the system, could be incorporated for estimating system reliability. Martz *et al* (1988) propose a two-stage Bayesian analysis, the first stage completed at the subsystem-level and the second stage at the system-level. To perform the subsystem-level analysis, each basic component is considered in turn and a Bayesian analysis is performed to obtain the posterior distribution for the reliability of each component.

Then, for the i^{th} subsystem, the posterior distributions for the components comprising subsystem i are combined to create an *induced* prior distribution for the reliability of subsystem i. If any historical data are available about subsystem i's reliability, it is used to form the *native* prior distribution. Together, the induced and native Subsystem i reliability prior distributions form the *combined* prior distribution for the reliability of Subsystem i. The combined prior distribution and any available data are used to obtain the posterior distribution for the reliability of Subsystem i. The first stage of the analysis is complete when all subsystems have been considered. The system level analysis is performed similarly. Martz and Waller (1990) extend this model to accommodate identical components in which the data for a set of common components comes from an identical generic component.

The work of Martz *et al* (1988) and Martz and Waller (1990) occurred before use of Markov chain Monte Carlo (MCMC) became mainstream and thus employs various approximations to obtain the system and subsystem reliability posterior distributions. In addition, component reliability assessments do not use any of the information available at higher levels of the system; for example, the prior and posterior distributions for the reliability of basic components do not include any information available from subsystem and system component historical data while the combined prior and posterior distributions for the reliability of subsystems do not utilize any of the historical data available at the system level. Johnson *et al* (2003) introduce a fully Bayesian approach for estimating system reliability when binomial data and expert information are available at various system levels. Their model does not require the use of analytical approximations, as it can be implemented via MCMC (Robert and Casella (2004), Gelman *et al* (2004)). Further, their model allows for the sharing of information between low-level and higher-level system components. The model of Johnson *et al* (2003) will serve as our framework for assessing the reliability of systems and is described in the following section.

1.2 A Hierarchical Model for Combining Multiple Information Sources to Assess System Reliability

Johnson *et al* (2003) introduce a Bayesian hierarchical model that utilizes data from

multiple sources to estimate the reliability of a system. This model accommodates binomial data at any level of the system, as well as an expert's "best guess" about component reliabilities. The full model as described in Johnson *et al* (2003) allows for another type of expert judgment by incorporating the possibility of grouping together components with similar reliabilities; we omit this type of expert information from the remaining discussion.

The model of Johnson *et al* (2003) assumes that each component test results in either a success or failure, where p_i denotes the reliability, or the probability of success, for Component i. In this model, the reliability for all non-basic components is written in terms of basic components. For example, consider the event tree for the missile system (Figure 1.3) from Johnson *et al* (2003). The subsystem labeled Component 14 is formed by Components 1 - 8 connected in series; the reliability of Component 14 is expressed as

$$p_{14} = \prod_{i=1}^{8} p_i.$$

Assuming that tests on a component are performed independently, and independent of tests performed on another component, the likelihood is proportional to

$$f(\mathbf{x}|\mathbf{p}) \propto \prod_{S_0} p_i^{x_i} \left(1 - p_i\right)^{n_i - x_i},$$

where S_0 is the set of components for which binomial data are available, n_i is the number of tests performed on Component i, and x_i is the number of successful Component i tests.

Johnson *et al* (2003) make an exchangeability assumption on the basic component reliabilities. This is modeled by assuming that the basic component reliabilities are drawn at random from a common distribution. Johnson *et al* (2003) specify this prior distribution to be Beta($J\gamma$, $J(1-\gamma)$). The mode of the prior distribution is γ while J is inversely proportional to the variance of the prior distribution (Johnson *et al* (2003)). The parameters J and γ are assigned the prior distributions Gamma(τ, ϕ) and Beta(ψ, ω), respectively. Johnson *et al* (2003) and Anderson-Cook *et al* (2007) use a Gamma(5, 1) prior distribution for J and the Jeffrey's prior distribution, Beta($\frac{1}{2}$, $\frac{1}{2}$), for γ in the stockpile reliability application. An important distinction is that they parameterize the Gamma distribution such that the mean

is $\mu = \frac{\tau}{\phi}$ and the variance is $\sigma^2 = \frac{\tau}{\phi^2}$ and thus the probability density function is given by

$$G(J;\tau,\phi) = \frac{\phi^\tau}{\Gamma(\tau)} J^{\tau-1}\exp(\phi J).$$

The model allows further prior information to be incorporated in the form of one or more experts' best guesses about a particular component's reliability. Specifically, Expert m specifies a value $\pi_{i,m}$ for the reliability of Component i; the value specified here can come from historical data or other intuition the expert has about the component's reliability. Johnson *et al* (2003) model this prior information about p_i as Beta($N_m\pi_{i,m}+1$, $N_m(1-\pi_{i,m})+1$). Anderson-Cook *et al* (2007) note that the form of the prior contribution due to Expert m's best guess $\pi_{i,m}$ has the form of a binomial likelihood with sample size N_m,

$$g(p_i|N_m,\pi_{i,m}) \propto p_i^{N_m\pi_{i,m}}\left(1-p_i\right)^{N_m(1-\pi_{i,m})}.$$

The parameter N_m is a precision parameter corresponding to the degree of belief in Expert m's best guesses (Anderson-Cook *et al* (2007)). The value of N_m for each expert is assumed to be drawn from Gamma(α_m, β_m). In the stockpile reliability application, Johnson *et al* (2003) and Anderson-Cook *et al* (2007) use a Gamma(5,1) prior distribution; this choice of prior distribution says, on average, the expert's guess is worth five tests (Hamada *et al* (2008)). Anderson-Cook *et al* (2007) note that if the expert's best guess contradicts the data at hand, the posterior distribution of N_m for that expert will tend put weight on small values, and essentially downweight Expert m's contribution.

Combining the priors and likelihood, the joint posterior distribution of the model parameters is proportional to

$$\begin{aligned} g(\mathbf{p},\boldsymbol{N},\gamma,J|\mathbf{x},\mathbf{n},\boldsymbol{\pi},\boldsymbol{\alpha},\boldsymbol{\beta},\psi,\omega,\tau,\phi) \quad \propto \quad & \prod_{i\in S_0} p_i^{x_i}(1-p_i)^{n_i-x_i} \\ \times \quad & \prod_{(i,m)\in S_1} B(p_i; N_m\pi_{i,m}+1,\ N_m(1-\pi_{i,m})+1) \\ \times \quad & \prod_{m:(i,m)\in S_1} G(N_m;\alpha_m,\ \beta_m) \\ \times \quad & \prod_{i\in S_2} B(p_i; J\gamma,\ J(1-\gamma)) \\ \times \quad & B(\gamma;\psi,\ \omega)G(J;\tau,\ \phi), \end{aligned} \tag{1.1}$$

where S_1 is the set of pairs (i, m) such that Expert m has specified a best guess $\pi_{i,m}$ about the reliability of Component i and S_2 is the set of basic components.

To estimate the model parameters and perform inference, Johnson *et al* (2003) and Anderson-Cook *et al* (2007) use a component-wise random-walk Metropolis-Hastings algorithm implemented in YADAS (Graves (2001), Graves (2003)). In this implementation, the basic component reliabilities and γ are updated by drawing Gaussian proposals on the logistic scale, while the precision parameters, J and N_m, are updated by Gaussian proposals specified on the logarithmic scale.

Johnson *et al* (2003) also describe how the model can be modified slightly to accommodate degradation models. The model of Johnson *et al* (2003) assumes that all tests performed on the components are independent of one another. Graves *et al* (2008) introduce a technique for analyzing simultaneous higher-level and partial lower-level data; that is, they address the case that tests performed on higher-level components provide partial information about the success of lower-level system components.

1.3 Implementation

Like Johnson *et al* (2003) and Anderson-Cook *et al* (2007), we use the Metropolis-Hastings algorithm to sample from the joint posterior distribution in Equation 1.1 (Robert and Casella (2004), Gelman *et al* (2004)). In this section we discuss several key aspects of our implementation. We first address the representation of the system. To prevent the computer code used to implement the Metropolis-Hastings algorithm from being system-specific, we describe our general "design matrix" system representation that easily allows the MCMC program to handle series, parallel, and series/parallel systems. We also provide the details of our implementation of the Metropolis-Hastings algorithm, including the implementation of automatic step size tuning (Graves (2005)).

1.3.1 System Representation

To allow the Metropolis-Hastings program to work with all types of systems, we need a flexible, generic system representation; we use a "design matrix" representation for the system. For a k-component system, the design matrix consists of k rows and $k-1$ columns; there is a row for every component and a column for all components except the system itself. Let $a_{i,j}$ denote the entry in the i^{th} row and the j^{th} column of the design matrix; the i^{th} row of the design matrix corresponds to Component i while the columns indicate which components comprise Component i. If Component i is a basic component, $a_{i,i} = 1$ and $a_{i,j} = 0$, for all $j \neq i$; for example, if Component 1 is a basic component, there will be a "1" in the first column in Row 1 and zeros at all other entries in Row 1. For non-basic components, $a_{i,j} = 0$ if Component i does not contain Component j. If Component i does contain Component j, $a_{i,j}$ indicates the connection type with either $a_{i,j} = 1$ if the connection is series or $a_{i,j} = 2$ if the connection is parallel. Thus rows corresponding to a non-basic component will either consist of multiple 1's if the components comprising the non-basic component are connected in series or multiple 2's if they are connected in parallel.

With this design matrix representation, each basic component and subsystem must have a row defining it before it can be used to define the system or another subsystem. Once all system components have been specified in the design matrix, the non-basic component reliabilities can easily be expressed using the following algorithm.

1. Identify the rows corresponding to basic components; the reliability of basic component i is p_i.

2. Identify the connection type for the rows corresponding to non-basic components; multiple 1's in a row indicate components connected in series to form a subsystem while multiple 2's in a row indicate a parallel connection.

3. Express the reliability of the non-basic components in terms of basic components:

   (a) If the connection type for Component i is series (1), $p_i = \prod p_j^{I(a_{i,j}=1)}$.

   (b) If the connection type for Component i is parallel (2), $p_i = 1 - \prod (1-p_j)^{I(a_{i,j}=2)}$.

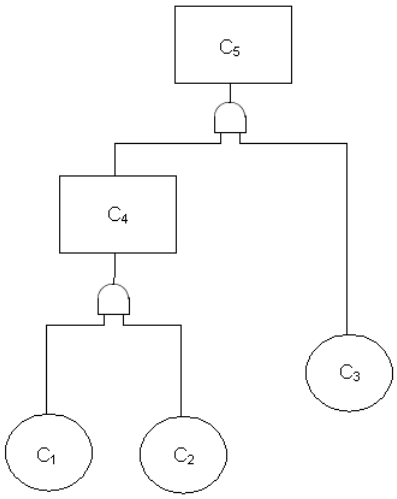


Figure 1.4 Event tree for five component series system

As an example, consider the event tree for a five component series system displayed in Figure 1.4. The design matrix representation of this system is

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}.$$

With this notation, the reliabilities of Components 1, 2, and 3 are p_1, p_2, and p_3, respectively, while the reliability of Component 4 is $p_4 = p_1 p_2$ and the reliability of Component 5 is $p_5 = p_3 p_4 = p_1 p_2 p_3$.

If Components 1 and 2 were connected in parallel, rather than in series, to form Component 4, the design matrix representation of the system would be

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 2 & 2 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}.$$

The reliabilities of Components 1, 2, and 3 are still p_1, p_2, and p_3, respectively, but now the reliability of Component 4 is $p_4 = 1 - (1 - p_1)(1 - p_2)$ and the reliability of Component 5 is $p_5 = p_3 p_4 = p_3 \left(1 - (1 - p_1)(1 - p_2)\right)$.

1.3.2 Random-Walk Metropolis-Hastings

To perform the analysis described Section 1.2 we implement a component-wise random-walk Metropolis-Hastings algorithm, similar to that of Johnson *et al* (2003), where the parameter vector $(\mathbf{p}, \boldsymbol{N}, J, \gamma)$ is divided into four components, or blocks. Each block is updated using a random-walk. A random walk takes into account the previously drawn value to simulate the next value; that is, given that the current draw at iteration t is $\theta^{(t)}$ the next value is generated according to $\theta^{(t+1)} = \theta^{(t)} + \epsilon_t$, where ϵ_t is drawn from a proposal distribution with mean 0 and standard deviation σ_ϵ, independent of $\theta^{(t)}$ (Robert and Casella (2004)). In our implementation, the block involving basic component reliabilities is updated with Gaussian proposals on the logistic scale. Once proposals are drawn for the basic components, the corresponding draws for non-basic components are computed according to the specified system design matrix, as described in Section 1.3.1. Similarly, the block consisting of the parameter γ is updated by drawing Gaussian proposals on the logistic scale. The blocks corresponding to the precision parameters, $\boldsymbol{N}$ and J, are updated by drawing Gaussian proposals on the log scale.

An important detail in implementing a random-walk is that of choosing the appropriate standard deviation for the proposal distribution, thus deciding the typical step size of the walk. Robert and Casella (2004) note that in the random-walk version of Metropolis-Hastings a high acceptance rate does not necessarily indicate that the algorithm is moving correctly; it may instead indicate that the step size, as determined by the standard deviation of the proposal distribution is too small, causing the chain to move slowly around the space. They also note that a small acceptance rate may indicate that the algorithm is moving across the space too quickly (i.e., with a step size that is too large). Roberts and Rosenthal (2001) conclude that an acceptance rate between 0.15 and 0.5 leads to an efficient random-walk Metropolis-Hastings algorithm. Graves (2005) describes a clever approach to automatically tune the step size via

logistic regression. To do this, a collection of logarithmically spaced step sizes are collected and the acceptance rates for the various step sizes are monitored. Let π_s denote the acceptance rate with step size s. The logistic regression

$$\text{logit}\,(\pi_s) = a + b\,\log(s)$$

is then fit to these "data" (Graves (2005)); once parameter estimates have been obtained, Graves (2005) uses them to predict the step size necessary to achieve the desired acceptance rate. The Newton-Raphson method is used to find the parameter estimates in the logistic regression, the details of which follow.

To implement this tuning technique, we use a sequence of 14 step sizes (i.e., $\exp((-6+k)/2)$, $k = 0, 1, \ldots, 13$). Each step size is in turn used as the standard deviation for the proposal distribution for each block of parameters (this tuning is done simultaneously for the four parameter blocks) and 100 posterior draws are collected. After posterior draws for the four blocks of parameters have been collected using all 14 step sizes, the step size necessary to achieve the target acceptance rate is predicted for each of the parameter blocks as follows.

Let s_i represent the i^{th} step size, n_i be the number of posterior draws, and x_i the number of accepted values in the n_i draws for the current parameter block. Suppose $\hat{a}^{(t)}$ and $\hat{b}^{(t)}$ are the parameter estimates after the t^{th} iteration of the Newton-Raphson algorithm; the initial values for a and b are specified so that $a^{(0)} + b^{(0)} = 0$ (Graves (2005)). After t iterations, the estimated acceptance rate for the current parameter block using step size s_i is

$$\hat{p}_i^t = \frac{\exp\left(\hat{a}^{(t)} + \hat{b}^{(t)}\log(s_i)\right)}{1 + \exp\left(\hat{a}^{(t)} + \hat{b}^{(t)}\log(s_i)\right)}.$$

Let $\hat{\beta}^{(t)} = (\hat{a}^{(t)}, \hat{b}^{(t)})'$, the Newton-Raphson equations are given by

$$\hat{\beta}^{(t+1)} = \hat{\beta}^{(t)} + \left(\mathbf{X}'\mathbf{V}\mathbf{X}\right)^{-1}\mathbf{X}'(\mathbf{x} - \boldsymbol{\mu}),$$

where $\mathbf{V} = diag\left(n_i\hat{p}_i^{(t)}(1-\hat{p}_i^{(t)})\right)$,

$$\mathbf{X} = \begin{bmatrix} 1 & \log(s_1) \\ 1 & \log(s_2) \\ \vdots & \vdots \\ 1 & \log(s_N) \end{bmatrix},$$

$\mathbf{x} = (x_1, x_2, \ldots, x_N)'$, and $\boldsymbol{\mu} = (n_1\hat{p}_1^{(t)}, n_2\hat{p}_2^{(t)}, \ldots, n_N\hat{p}_N^{(t)})$ (Agresti (2002)).

At iteration $t+1$ the parameter estimates of a and b are given by,

$$\hat{a}^{(t+1)} = \hat{a}^{(t)} + \frac{C^{(t)}D^{(t)} - B^{(t)}E^{(t)}}{A^{(t)}C^{(t)} - B^{(t)^2}} \tag{1.2}$$

and

$$\hat{b}^{(t+1)} = \hat{b}^{(t)} + \frac{A^{(t)}E^{(t)} - B^{(t)}D^{(t)}}{A^{(t)}C^{(t)} - B^{(t)^2}}, \tag{1.3}$$

where $A^{(t)} = \sum_i n_i\hat{p}_i^{(t)}(1-\hat{p}_i^{(t)})$, $B^{(t)} = \sum_i n_i\log(s_i)\hat{p}_i^{(t)}(1-\hat{p}_i^{(t)})$, $C^{(t)} = \sum_i n_i(\log s_i)^2\hat{p}_i^{(t)}(1-\hat{p}_i^{(t)})$, $D^{(t)} = \sum_i(x_i - n_i\hat{p}_i^{(t)})$, and $E^{(t)} = \sum_i \log(s_i)(x_i - n_i\hat{p}_i^{(t)})$.

The method converges when the parameter estimates do not change more than a specified amount. After convergence, the estimates ($\hat{a}$ and $\hat{b}$) can be used to predict the step size necessary to achieve the target acceptance rate for the current parameter block,

$$s_{target} = \exp\left\{\frac{\text{logit}(\pi_{target}) - \hat{a}}{\hat{b}}\right\}. \tag{1.4}$$

This process is repeated for the remaining parameter blocks to predict the step size necessary for that block to achieve the target acceptance rate.

The algorithm typically converges quite quickly, often after 6 - 8 iterations. After the tuning phase, the remaining posterior draws are made according to the target step size. Once the tuning phase is complete, the burn-in period begins; thus none of the draws made during the tuning phase are kept as part of the simulated posterior. After the burn-in period, the posterior draws are also thinned to account for correlation between consecutive draws.

1.4 Introduction to Resource Allocation

After estimating system reliability based on first-stage data, the next issue is to plan how to collect second-stage data to provide further information about the system's reliability; this problem is often referred to as "resource allocation" (Hamada *et al* (2004), Wilson *et al* (2006), Hamada *et al* (2008)). Under the resource allocation framework, there is some (limited) amount of resources available to collect new data. The various components that comprise the system have inherent testing costs associated with them; the issue is to decide with these resources, how many tests should be performed on each component to achieve the most new information about the reliability of the system.

One issue to address is the choice of the planning criterion, which measures the expected amount of information to be gained by performing a specific experiment. Previous resource allocation studies (Hamada *et al* (2004), Wilson *et al* (2006), and Hamada *et al* (2008)) use the width of a central credible interval to measure the expected amount of information to be gained from an experiment. In Chapter 2 we propose the use of entropy as the planning criterion.

The strategy employed by Hamada *et al* (2004), Wilson *et al* (2006), and Hamada *et al* (2008) for finding an optimal resource allocation is one based on generating multiple data sets consistent with the first-stage data and current model parameter estimates. These simulated data sets are used for repeated pre-posterior analyses (i.e., these analyses are done before the second-stage data is collected); the current approach is described fully in Chapter 2. A disadvantage of this approach is that the repeated analyses are done under the framework of the model described in Section 1.2 and thus are time-consuming and computationally intensive. Wilson *et al* (2006) recognize these problems:

> "... system reliability assessments are computationally intensive. What approximations can be incorporated without sacrificing accuracy? Or do we need the power of a supercomputer? Resource allocation is even more computationally intensive and brings the issues of computation to the forefront. (Wilson *et al* (2006))"

Our goal is to address the issues brought up by Wilson *et al* (2006) and reduce the computational burden associated with finding optimal resource allocations. In Chapter 2 we introduce new, computationally efficient methodology for evaluating candidate allocations. This new methodology does not require repeated analyses via MCMC; multiple candidate allocations can be evaluated based on the results of the initial analysis of the first-stage data. The new approach works well for small to moderate size systems, and thus we introduce a modification of the methodology to handle larger systems. Examples using both methodologies (for small and large systems) are provided. The approach introduced in Chapter 2 uses a histogram approximation of a continuous distribution; the error induced by this approximation is investigated in the case that the continuous distribution is a Beta distribution.

Because of the ease with which the new methodology can evaluate multiple candidate allocations, it works well when coupled with an optimization procedure. In Chapter 3 we introduce genetic algorithms (GAs) and their application to finding optimal resource allocations. We describe an implementation of GAs that uses the candidate evaluation methodology from Chapter 2. Examples of the GA implementation for systems of varying size are provided.

Chapter 4 investigates two real systems and how the methodology of Chapters 2 and 3 can be applied. The first case study considers the reliability of a missile system studied by Martz *et al* (1988). This missile system is a series system consisting of five major subsystems. Binomial data are available for three of these subsystems and their basic components, while a number of the components have available historical data. We analyze the initial data using the approach of Johnson *et al* (2003) and then use the methodology in Chapters 2 and 3 to find optimal resource allocations under different hypothetical cost structures and budgets. The second case study addresses the series/parallel system described by Martz and Waller (1990); the intent of the initial study was to estimate the demand-unavailability of one of the safety features in a certain 1,150 megawatt electric U.S. commerical nuclear-power boiling-water reactor. We again analyze the existing binomial data using the model of Johnson *et al* (2003) and find optimal resource allocations under various hypothetical cost structures and budgets. In Chapter 5 we summarize our findings and present ideas for related future work.

CHAPTER 2. EVALUATION OF CANDIDATE ALLOCATIONS

In data collection planning for system reliability studies, the goal is to decide which components, and how many of each, will yield the most new information about the reliability of the system in question. We would like to use first-stage data and information from previous studies to decide which allocation of resources we expect to be most informative. Intuitively, components that are critical to the system's success, have some uncertainty based on the available first-stage data and prior information, and/or moderate testing costs will likely be selected for collecting second-stage data, whereas components that are not highly critical for system success, have high first-stage success rates, and/or are expensive to test are less likely to be included in the second-stage experiment.

Ultimately we would like to find an optimal, or nearly optimal, allocation; that is, the allocation that yields the most new information about the system's reliability for the resources available. In order to do this, we must be able to evaluate how much information we expect to gain from a candidate experiment, or allocation. To do this, we first address the choice of *planning criterion* (Hamada *et al* (2008)); the planning criterion is used to measure the amount of information provided by a candidate allocation. Once a planning criterion has been selected, the issue then becomes the computation of that criterion to evaluate a candidate allocation. We provide an overview of the approach used, for example, by Hamada *et al* (2004); we refer to this as the "current approach" throughout this dissertation. The current approach is computationally intensive, requiring repeated pre-posterior analyses (i.e., analyses performed before any second-stage data are actually collected (Hamada *et al* (2008))) via MCMC. New, computationally efficient methodology for evaluating candidate allocations is introduced that allows evaluation of multiple candidate allocations based on the results from a single analysis

of the first-stage data via MCMC. This new methodology requires a trade-off between computation time and a large enumeration problem for some candidate allocations, and thus a sampling-based modification is described for larger applications.

2.1 Expected Information Gain

In order to evaluate a candidate allocation, we must first choose how to measure information gain; Hamada *et al* (2008) refer to this as the *planning criterion*. Our primary interest lies in the estimation of the system reliability, which we will denote as θ. In this multi-stage data collection process, we will use $\mathbf{n}_1 = (n_{1,1}, n_{1,2}, \ldots, n_{1,k}, n_{1,\theta})$ to denote the number of tests performed on each component during the initial testing phase, where $n_{1,i}$ denotes the number of initial tests performed on the system components, $i = 1, \ldots, k$, and $n_{1,\theta}$ denotes the number of initial tests performed on the full system. The vector $\mathbf{x}_1 = (x_{1,1}, x_{1,2}, \ldots, x_{1,k}, x_{1,\theta})$ represents the number of initial successes observed for each component during the initial testing phase. A candidate allocation will be represented as $\mathbf{n}_2 = (n_{2,1}, n_{2,2}, \ldots, n_{2,k}, n_{2,\theta})$ with an outcome from this allocation denoted as $\mathbf{x}_2 = (x_{2,1}, x_{2,2}, \ldots, x_{2,k}, x_{2,\theta})$. Throughout our discussion, we regard $\mathbf{x}_1$ as known and constant, whereas $\mathbf{x}_2$ represents the hypothetical outcome of the second-stage test specified by $\mathbf{n}_2$.

Once an experiment is performed, the amount of information we have about the system's reliability should be increased. That is, there should be less uncertainty in the system reliability posterior distribution than there was prior to the second-stage experiment being performed. Ideally, we would like to perform the experiment (i.e., select the vector $\mathbf{n}_2$) that provides the most gain in information and thus the largest reduction in uncertainty. In previous resource allocation studies, the expected (over values of the unobserved second-stage data) width of a central credible set has been used as the planning criterion (Hamada *et al* (2004), Wilson *et al* (2006), Hamada *et al* (2008)); that is, an experiment that provides a large expected reduction in the width of a central, say 90%, credible set is chosen for collecting second-stage data.

Alternatively, a Bayesian experimental design approach could be used. Under this ap-

proach, a utility function that reflects the goals of the experiment is chosen (Chaloner and Verdinelli (1995)). The optimal experimental design would be the design that maximizes the expected utility. For example, when the purpose of the study is parameter estimation, a utility function based on Shannon information is often used. The optimal design would then be the design that maximizes the expected gain in Shannon information (Lindley (1956), Chaloner and Verdinelli (1995)); this is equivalent to maximizing the expected Kullback-Leibler distance between the prior and posterior distributions (Chaloner and Verdinelli (1995), Hamada *et al* (2001)). Under the resource allocation framework, this is the design that maximizes

$$\sum_{\mathbf{x}_2} \int \log \left\{ \frac{f(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2, \mathbf{x}_2)}{f(\theta|\mathbf{n}_1, \mathbf{x}_1)} \right\} f(\mathbf{x}_2, \theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2) d\theta, \tag{2.1}$$

where $f(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2, \mathbf{x}_2)$ is the updated system reliability posterior distribution given the first-stage data and a hypothetical outcome $\mathbf{x}_2$ from the candidate allocation $\mathbf{n}_2$ and $f(\theta|\mathbf{n}_1, \mathbf{x}_1)$ is the system reliability posterior distribution based on the first-stage data, which does not depend the allocation $\mathbf{n}_2$. Thus (2.1) can be rewritten as

$$\sum_{\mathbf{x}_2} \int \log \left(f(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2, \mathbf{x}_2) \right) f(\mathbf{x}_2, \theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2) d\theta \quad - \quad \int \log \left(f(\theta|\mathbf{n}_1, \mathbf{x}_1) \right) f(\theta|\mathbf{n}_1, \mathbf{x}_1) d\theta; \tag{2.2}$$

the second term in (2.2) does not depend on the allocation $\mathbf{n}_2$, and thus maximizing (2.1) is equivalent to maximizing the expected Shannon information of the system reliability posterior distribution with respect to selection of $\mathbf{n}_2$ (i.e., the first term in (2.2)).

Further, note that the first term in (2.2) can be rewritten as

$$\begin{aligned} -\sum_{\mathbf{x}_2} f(\mathbf{x}_2|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2) \left[- \int f(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2, \mathbf{x}_2) \log \left(f(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2, \mathbf{x}_2) \right) d\theta \right] \\ = -E_{\mathbf{x}_2|\mathbf{x}_1} H(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2, \mathbf{x}_2), \end{aligned} \tag{2.3}$$

where

$$H(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2, \mathbf{x}_2) \quad = \quad - \int f(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2, \mathbf{x}_2) \log \left(f(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2, \mathbf{x}_2) \right) d\theta \tag{2.4}$$

is the entropy of the updated system reliability posterior distribution. Thus, maximizing the expected Shannon information of the posterior distribution is equivalent to minimizing the expected entropy of the updated posterior distribution. We choose to use the expected entropy of the updated system reliability posterior distribution as our planning criterion; we

want to find an allocation that minimizes the expected entropy of the updated system reliability posterior distribution.

Intuitively, entropy is a measure of the dispersion of a distribution, with high entropy indicating a more widely dispersed distribution (Cover and Thomas (1991)). Table 2.1 illustrates the entropy for various Beta densities, while the densities are displayed in Figure 2.1; the Beta distributions considered have fixed $\alpha = 10$ and varying β parameter. The most "peaked" density has the lowest entropy while the most widely spread density has the highest entropy. The Beta(1, 1) distribution has entropy 0, and the Beta(α, β) and Beta(β, α) distributions have the same entropy, as they have the same amount of dispersion. For continuous densities, entropy can take on any real value; for discrete distributions, entropy is strictly non-negative.

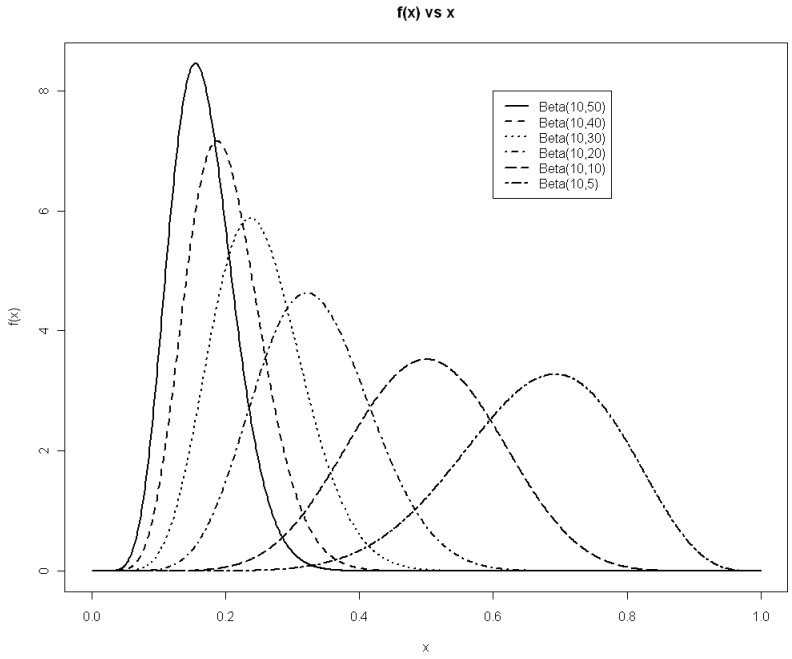


Figure 2.1 Probability density functions for Beta distributions with $\alpha = 10$, β=5, 10, 20, 30, 40, and 50

It is not possible to analytically find the resource allocation that yields the smallest expected entropy in the problem described here; the form of the updated posterior is not immediately obvious and further it depends on second-stage data that are yet to be observed. The following sections describe different strategies for evaluating the planning criterion associated with

candidate allocations.

Table 2.1 Entropy for Beta distributions with $\alpha = 10$, β=5, 10, 20, 30, 40, and 50

β	**Entropy**
50	-1.642216
40	-1.479193
30	-1.286933
20	-1.056789
10	-0.798344
5	-0.7350772

2.2 Current Approach for Comparing Candidate Allocations

Hamada *et al* (2004), Wilson *et al* (2006), and Hamada *et al* (2008) use a simulation-based approach to evaluate the planning criterion associated with candidate allocations. First, they perform a Bayesian analysis to obtain the system reliability posterior distribution given the initial data, $f(\theta|\mathbf{n}_1, \mathbf{x}_1)$; the complexity of the model requires the posterior distribution be explored via MCMC. Next, for a given candidate allocation $\mathbf{n}_2$, they draw values for the component reliabilities from the joint posterior distribution given the initial data and use these draws to generate second-stage data $\mathbf{x}_2$ from the candidate allocation $\mathbf{n}_2$. They then perform a new analysis using the initial data augmented with the new data to obtain the updated system reliability posterior distribution given the initial and new data, $f(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2, \mathbf{x}_2)$. Once they obtain draws from the updated system reliability posterior distribution, they compute the planning criterion; their planning criterion is generally the width of the central 90% credible interval for system reliability, but they could easily use the entropy of the updated system reliability posterior distribution.

This planning criterion value is based on only a single realization ($\mathbf{x}_2$) from the candidate allocation. To consider additional realizations, they repeat this process r times, each based on an independently generated outcome for the candidate allocation, and then obtain the empirical distribution of their planning criterion for the current candidate allocation. They report a typical value for the planning criterion to be an upper quantile (i.e., 90%) of this

empirical distribution.

They must repeat this entire process for any other candidate allocations they wish to consider. Thus, to evaluate the planning criterion associated with each of c candidate allocations using this simulation-based approach, it is necessary to generate $c \times r$ new datasets and perform $c \times r$ analyses via MCMC. To select the most informative of the c candidate allocations evaluated, they choose the allocation yielding the best value of the planning criterion (i.e., the candidate allocation they expect to yield the narrowest credible interval). In Section 2.3 we describe a more computationally efficient approach for evaluating candidate allocations that, for problems of limited size, can be executed on the information from a single MCMC analysis of the initial data.

2.3 Proposed Approach for Comparing Candidate Allocations

The current approach for evaluating candidate allocations requires multiple runs of an MCMC algorithm to compute the empirical distribution of the planning criterion, from which a typical amount of information gain from a candidate allocation can be calculated; this procedure needs to be repeated for every candidate to be considered. Here we propose an approach that can be used to evaluate the planning criteria multiple candidate allocations based solely on the analysis of the initial data, thus requiring only a single call to an MCMC algorithm. The key to this approach lies in utilizing the relationship between the two posterior distributions, $f(\theta|\mathbf{n}_1, \mathbf{x}_1)$ and $f(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2, \mathbf{x}_2)$, and the pass/fail nature of the tests comprising a candidate allocation. For notational convenience, dependence on the allocations $\mathbf{n}_1$ and $\mathbf{n}_2$ will generally be suppressed for the remainder of this dissertation; e.g., $f(\theta|\mathbf{n}_1, \mathbf{x}_1)$ and $f(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2, \mathbf{x}_2)$ will now be denoted as $f(\theta|\mathbf{x}_1)$ and $f(\theta|\mathbf{x}_1, \mathbf{x}_2)$, respectively. For a given outcome $\mathbf{x}_2$ from a candidate allocation $\mathbf{n}_2$, the updated posterior distribution of system reliability given the initial data and the outcome from the candidate allocation can be written as

$$\begin{aligned} f(\theta|\mathbf{x}_1, \mathbf{x}_2) &= \frac{f(\theta, \mathbf{x}_1, \mathbf{x}_2)}{p(\mathbf{x}_1, \mathbf{x}_2)} \\ &= \frac{p(\mathbf{x}_2|\theta) f(\theta|\mathbf{x}_1)}{p(\mathbf{x}_2|\mathbf{x}_1)}, \end{aligned} \tag{2.5}$$

since $p(\mathbf{x}_2|\theta, \mathbf{x}_1) = p(\mathbf{x}_2|\theta)$, (i.e., $\mathbf{x}_1$ and $\mathbf{x}_2$ are conditionally independent given θ) (Gelman *et al* (2004)).

The three factors in Equation 2.5 can be computed using the output from the initial analysis obtained via a single run of an MCMC algorithm and combined to compute the updated posterior distribution for a given outcome $\mathbf{x}_2$. To do this, we use the initial set of data to estimate the effect that new data will have on the updated system reliability posterior distribution. The initial analysis yields posterior draws for the model parameters, including component reliabilities (p_1, ..., p_k), system reliability (θ) and the other model parameters (e.g., the epxert precision parameters, J, and γ); Table 2.2 displays what this may look like for M draws from the joint first-stage posterior distribution. Once this initial analysis has been performed, we can calculate the amount of uncertainty we would "expect" to see if we performed any candidate experiment.

After the initial analysis has been performed, we determine the conditional distribution (given $\mathbf{x}_1$) for new test data. To do this, we use the fact that a candidate allocation, $\mathbf{n}_2 = (n_{2,1}, n_{2,2}, \ldots, n_{2,k}, n_{2,\theta})$, is made up of independent pass/fail tests on the various components. Thus for each draw from the joint posterior distribution obtained in the initial analysis, we can compute the probability of each possible outcome for individual component (i.e., Components $1 - k$ and the full system) tests given the current reliability draw for that component. For example, if $n_{2,1}$ second-stage tests are to be performed on Component 1, the probability of $x_{2,1} = 0, 1, \ldots, n_{2,1}$ successes for Component 1 is Binomial with probability $p_1^{(1)}$ for the first draw from the posterior, Binomial with probability $p_1^{(2)}$ for the second posterior draw, etc. Similarly, conditional distributions (given $\mathbf{x}_1$) for second-stage data can be calculated for the

Table 2.2 Joint first-stage posterior draws from an MCMC sampler

p_1	$\cdots$	p_k	θ	$\cdots$
$p_1^{(1)}$	$\cdots$	$p_k^{(1)}$	$\theta^{(1)}$	$\cdots$
$p_1^{(2)}$	$\cdots$	$p_k^{(2)}$	$\theta^{(2)}$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$p_1^{(M)}$	$\cdots$	$p_k^{(M)}$	$\theta^{(M)}$	$\cdots$

Table 2.3 System reliability posterior draws and computed outcome probabilities for a candidate allocation $\mathbf{n}_2$

θ	$p(\mathbf{x}_2 = \mathbf{0}\|\mathbf{n}_2, p_1, \ldots, p_k, \theta)$	$\cdots$	$p(\mathbf{x}_2 = \mathbf{n}_2\|\mathbf{n}_2, p_1, \ldots, p_k, \theta)$
$\theta^{(1)}$	$p(\mathbf{x}_2 = \mathbf{0}\|\mathbf{n}_2, p_1^{(1)}, \ldots, p_k^{(1)}, \theta^{(1)})$	$\cdots$	$p(\mathbf{x}_2 = \mathbf{n}_2\|\mathbf{n}_2, p_1^{(1)}, \ldots, p_k^{(1)}, \theta^{(1)})$
$\theta^{(2)}$	$p(\mathbf{x}_2 = \mathbf{0}\|\mathbf{n}_2, p_1^{(2)}, \ldots, p_k^{(2)}, \theta^{(2)})$	$\cdots$	$p(\mathbf{x}_2 = \mathbf{n}_2\|\mathbf{n}_2, p_1^{(2)}, \ldots, p_k^{(2)}, \theta^{(2)})$
$\vdots$	$\vdots$	$\cdots$	$\vdots$
$\theta^{(M)}$	$p(\mathbf{x}_2 = \mathbf{0}\|\mathbf{n}_2, p_1^{(M)}, \ldots, p_k^{(M)}, \theta^{(M)})$	$\cdots$	$p(\mathbf{x}_2 = \mathbf{n}_2\|\mathbf{n}_2, p_1^{(M)}, \ldots, p_k^{(M)}, \theta^{(M)})$

remaining system components for which there are second-stage tests.

The actual outcomes for the candidate allocation, $\mathbf{x}_2 = (x_{2,1}, x_{2,2}, \ldots, x_{2,\theta})$, are the combinations of outcomes for tests on individual components. If we assume that tests on components are independent of one another, the probability of a given outcome from the candidate allocation is the product of the Binomial probabilities for the number of successes from each individual component's tests. If the candidate allocation is $\mathbf{n}_2 = (n_{2,1}, n_{2,2}, \ldots, n_{2,k}, n_{2,\theta})$, then the probability of an outcome $\mathbf{x}_2 = (x_{2,1}, x_{2,2}, \ldots, x_{2,k}, x_{2,\theta})$ given the l^{th} set of parameter draws is

$$\begin{aligned} p\left(\mathbf{x}_2|\mathbf{n}_2, p_1^{(l)}, p_2^{(l)}, \ldots, p_k^{(l)}, \theta^{(l)}\right) &= p\left(x_{2,1}|n_{2,1}, p_1^{(l)}\right) p\left(x_{2,2}|n_{2,2}, p_2^{(l)}\right) \times \cdots \\ &\times \; p\left(x_{2,k}|n_{2,k}, p_k^{(l)}\right) p\left(x_{2,\theta}|n_{2,\theta}, \theta^{(l)}\right). \end{aligned} \tag{2.6}$$

Thus we can use the information from the initial analysis to find the probability of each hypothetical outcome $\mathbf{x}_2$ of a candidate allocation $\mathbf{n}_2$ for each draw from the joint posterior distribution. At this point, we are only interested in these outcome probabilities and the system reliability posterior draws from the initial analysis. Table 2.3 displays the form of the results for a given candidate allocation $\mathbf{n}_2$ after M draws from the posterior.

Table 2.3 contains all of the information necessary to calculate the three factors in Equation 2.5 (within the simulation accuracy associated with the value of M, the number of posterior draws). Throughout this dissertation, we use a $\tilde{}$ to denote a quantity calculated using the draws from the MCMC (i.e., the calculated updated system reliability posterior distribution will be denoted as $\tilde{f}(\theta|\mathbf{x}_1, \mathbf{x}_2)$). First consider $f(\theta|\mathbf{x}_1)$; this is the system reliability posterior distribution given only the first-stage data and will be the same for any candidate allocation

we wish to consider. To compute this piece, we focus on the θ column in the table displayed in Table 2.3. We compute the density of θ, $\tilde{f}(\theta|\mathbf{x}_1)$, using a histogram with bin width, h. Let B_j represent the j^{th} bin. If we denote the number of posterior draws that fall into the j^{th} bin as ν_j, the density at any θ in bin B_j is given by

$$\tilde{f}(\theta|\mathbf{x}_1) \quad = \quad \frac{\nu_j}{hM}, \text{ for } \theta \in B_j. \tag{2.7}$$

For a specific outcome $\mathbf{x}_2$, we calculate $p(\mathbf{x}_2|\theta)$ using the same bins defined to compute $\tilde{f}(\theta|\mathbf{x}_1)$, thus compute

$$\tilde{p}(\mathbf{x}_2|\theta) \quad = \quad \frac{\sum_{l=1}^{M} p(\mathbf{x}_2|\mathbf{n}_2, p_1^{(l)}, \ldots, p_k^{(l)}, \theta^{(l)}) I(\theta^{(l)} \in B_j)}{\nu_j}, \qquad \theta \in B_j. \tag{2.8}$$

To calculate $p(\mathbf{x}_2|\mathbf{x}_1)$ for a specific outcome $\mathbf{x}_2$, we average over the $p(\mathbf{x}_2|\mathbf{n}_2, p_1, \ldots, p_k, \theta)$ column of Table 2.3; that is,

$$\tilde{p}(\mathbf{x}_2|\mathbf{x}_1) \quad = \quad \frac{\sum_{l=1}^{M} p(\mathbf{x}_2|\mathbf{n}_2, p_1^{(l)}, \ldots, p_k^{(l)}, \theta^{(l)})}{M}. \tag{2.9}$$

Equations 2.7, 2.8, and 2.9 are combined to compute

$$\tilde{f}(\theta|\mathbf{x}_1, \mathbf{x}_2) \quad = \quad \frac{\tilde{f}(\theta|\mathbf{x}_1)\tilde{p}(\mathbf{x}_2|\theta)}{\tilde{p}(\mathbf{x}_2|\mathbf{x}_1)}, \tag{2.10}$$

for a discrete set of θ values and a given outcome $\mathbf{x}_2$. Next, we compute the entropy of the updated system reliability posterior distribution given the initial data and the hypothetical outcome $\mathbf{x}_2$ (Equation 2.4) as,

$$\tilde{H}(\theta|\mathbf{x}_1, \mathbf{x}_2) \quad = \quad -\sum_{\theta} h\tilde{f}(\theta|\mathbf{x}_1, \mathbf{x}_2) \log\left\{\tilde{f}(\theta|\mathbf{x}_1, \mathbf{x}_2)\right\}. \tag{2.11}$$

To verify that this is a reasonable way to calculate $H(\theta|\mathbf{x}_1, \mathbf{x}_2)$, we show that integrals of $\tilde{f}(\theta|\mathbf{x}_1, \mathbf{x}_2)$ over fixed intervals of θ approach the corresponding integrals of $f(\theta|\mathbf{x}_1, \mathbf{x}_2)$ as the number of posterior draws (M) increases and the bin width (h) decreases. First define a discrete random variable $\theta^{[\,]}$ with

$$\theta^{[\,]} = \theta^{[j]} \qquad \text{for } \theta \in B_j = ((j-1)h, jh],\ j = 1, \ldots, n_h = \lceil 1/h \rceil,$$

where $\lceil \frac{1}{h} \rceil$ is the number of bins used; this can be thought of as rounding the posterior system reliability draws to a specified number of decimal points. The random variable $\theta^{[\ ]}$ has probability mass function $p(\theta^{[\ ]}|\mathbf{x}_1)$. We calculate $p(\theta^{[j]}|\mathbf{x}_1)$ as the relative proportion of posterior draws contained in B_j; that is, let ν_j be the bin count for B_j, then

$$\begin{aligned} \tilde{p}(\theta^{[j]}|\mathbf{x}_1) &= \frac{\nu_j}{M} \\ &= \frac{1}{M}\sum_{l=1}^{M} I(\theta^l \in \theta^{[j]}). \end{aligned}$$

Note that from the MCMC draws we have a sample from $p(\theta^{[\ ]}|\mathbf{x}_1)$. Thus by the Law of Large Numbers (Casella and Berger (2002)),

$$\begin{aligned} \tilde{p}(\theta^{[j]}|\mathbf{x}_1) &\rightarrow E_{\theta^{[\ ]}} I(\theta^{[j]}) \\ &= \sum_{i=1}^{n_h} p(\theta^{[i]}|\mathbf{x}_1) I(\theta^{[j]}) \\ &= p(\theta^{[j]}|\mathbf{x}_1), \end{aligned}$$

as $M \rightarrow \infty$ and $\tilde{p}_h(\theta^{[j]}|\mathbf{x}_1) \equiv \frac{1}{h}\tilde{p}(\theta^{[j]}|\mathbf{x}_1) \rightarrow \frac{1}{h}p(\theta^{[j]}|\mathbf{x}_1) \equiv p_h(\theta^{[j]}|\mathbf{x}_1)$ as $M \rightarrow \infty$.

Next, recall

$$\tilde{p}(\mathbf{x}_2|\mathbf{x}_1) = \frac{1}{M}\sum_{l=1}^{M} p(\mathbf{x}_2|\mathbf{p}^{(l)}).$$

From the initial MCMC analysis, we have a sample from $f(\mathbf{p}|\mathbf{x}_1)$ and

$$\begin{aligned} p(\mathbf{x}_2|\mathbf{x}_1) &= \int p(\mathbf{x}_2, \mathbf{p}|\mathbf{x}_1) d\mathbf{p} \\ &= \int p(\mathbf{x}_2|\mathbf{p}) f(\mathbf{p}|\mathbf{x}_1) d\mathbf{p} \\ &= E_{\mathbf{p}|\mathbf{x}_1} p(\mathbf{x}_2|\mathbf{p}). \end{aligned}$$

Hence, by the Law of Large Numbers $\tilde{p}(\mathbf{x}_2|\mathbf{x}_1) \rightarrow E_{\mathbf{p}|\mathbf{x}_1} p(\mathbf{x}_2|\mathbf{p}) = p(\mathbf{x}_2|\mathbf{x}_1)$ as $M \rightarrow \infty$ (Casella and Berger (2002)).

Lastly,

$$\tilde{p}(\mathbf{x}_2|\theta^{[j]}) = \frac{1}{\nu_j}\sum_{l=1}^{M} p(\mathbf{x}_2|\mathbf{p}^{(l)}) I(\theta^{(l)} \in \theta^{[j]}).$$

By binning the system reliability draws, and corresponding draws for basic component reliabilities, from the MCMC we have a sample of size ν_j from $f(\mathbf{p})$ restricted to where $\theta(\mathbf{p}) \in \theta^{[j]}$

(where $\theta(\mathbf{p})$ is the system reliability computed from the basic component reliability posterior draws in $\mathbf{p}$). By the Law of Large Numbers,

$$\begin{aligned}\tilde{p}(\mathbf{x}_2|\theta^{[j]}) &\rightarrow \int_{\theta(\mathbf{p})\in\theta^{[j]}} p(\mathbf{x}_2|\mathbf{p})f(\mathbf{p}|\mathbf{x}_1)d\mathbf{p} \\ &= p(\mathbf{x}_2|\theta^{[j]}),\end{aligned}$$

as $M \rightarrow \infty$ and hence $\nu_j \rightarrow \infty$ (Casella and Berger (2002)).

Thus

$$\begin{aligned}\tilde{p}_h(\theta^{[j]}|\mathbf{x}_1,\mathbf{x}_2) \equiv \frac{\tilde{p}(\mathbf{x}_2|\theta^{[j]})\tilde{p}_h(\theta^{[j]}|\mathbf{x}_1)}{\tilde{p}(\mathbf{x}_2|\mathbf{x}_1)} &\rightarrow \frac{p(\mathbf{x}_2|\theta^{[j]})p_h(\theta^{[j]}|\mathbf{x}_1)}{p(\mathbf{x}_2|\mathbf{x}_1)} \\ &\equiv p_h(\theta^{[j]}|\mathbf{x}_1,\mathbf{x}_2) \\ &= \frac{1}{h}\int_{\theta\in\theta^{[j]}} f(\theta|\mathbf{x}_1,\mathbf{x}_2)d\theta \\ &= f(\theta^*|\mathbf{x}_1,\mathbf{x}_2), \qquad \text{for some } \theta^* \in \theta^{[j]}.\end{aligned}$$

The function $f(\theta)\log(f(\theta))$ is a smooth function in θ and thus

$$\begin{aligned}h\tilde{p}_h(\theta^{[j]}|\mathbf{x}_1,\mathbf{x}_2)\log\left(\tilde{p}_h(\theta^{[j]}|\mathbf{x}_1,\mathbf{x}_2)\right) &\rightarrow hf(\theta^*|\mathbf{x}_1,\mathbf{x}_2)\log\left(f(\theta^*|\mathbf{x}_1,\mathbf{x}_2)\right), \quad \text{for some } \theta^* \in \theta^{[j]} \\ &= \int_{\theta\in\theta^{[j]}} f(\theta|\mathbf{x}_1,\mathbf{x}_2)\log\left(f(\theta|\mathbf{x}_1,\mathbf{x}_2)\right)d\theta\end{aligned}$$

as $M \rightarrow \infty$, and hence

$$\sum_{j=1}^{n_h} h\tilde{p}_h(\theta^{[j]}|\mathbf{x}_1,\mathbf{x}_2)\log\left(\tilde{p}_h(\theta^{[j]}|\mathbf{x}_1,\mathbf{x}_2)\right) \rightarrow \int f(\theta|\mathbf{x}_1,\mathbf{x}_2)\log\left(f(\theta|\mathbf{x}_1,\mathbf{x}_2)\right)d\theta,$$

as $M \rightarrow \infty$ and $h \rightarrow 0$.

This approach can be used to calculate $H(\theta|\mathbf{x}_1,\mathbf{x}_2)$ for all possible outcomes $\mathbf{x}_2$ for a candidate allocation $\mathbf{n}_2$ using the information from the initial analysis via MCMC. The expected entropy for a candidate allocation $\mathbf{n}_2$ is then computed as

$$\begin{aligned}\tilde{G}(\theta;\mathbf{n}_1,\mathbf{x}_1,\mathbf{n}_2) &= E_{\mathbf{x}_2|\mathbf{x}_1}\tilde{H}(\theta|\mathbf{x}_1,\mathbf{x}_2) \\ &= \sum_{\mathbf{x}_2}\tilde{p}(\mathbf{x}_2|\mathbf{x}_1)\tilde{H}(\theta|\mathbf{x}_1,\mathbf{x}_2). \qquad (2.12)\end{aligned}$$

Further, the results of the initial analysis can be used to compute the expected entropy for any candidate allocation under consideration. Thus multiple candidate allocations can be evaluated using the results of a single MCMC analysis, greatly reducing the amount of computing time and power needed.

2.4 Implementation of Proposed Approach

The candidate evaluation procedure described in Section 2.3 is implemented in C. The evaluation program is provided the M posterior draws from the initial analysis, the specified bin width h, and a list of candidate allocations, $\mathbf{n}_2 = (n_{2,1}, n_{2,2}, \ldots, n_{2,k}, n_{2,\theta})$, to evaluate. The first step is to calculate $f(\theta|\mathbf{x}_1)$, which does not depend on the candidate under consideration and thus is calculated once at the beginning of the program call. In the implementation, $\tilde{f}(\theta|\mathbf{x}_1)$ is represented as a vector with length equal to the number of bins used in the histogram approximation (n_h). The j^{th} entry of the vector holds $\tilde{f}(\theta|\mathbf{x}_1)$, for $\theta \in B_j$; that is, the j^{th} entry in the vector, denoted here as $\tilde{f}_j(\theta|\mathbf{x}_1)$, holds

$$\tilde{f}_j(\theta|\mathbf{x}_1) = \frac{\nu_j}{Mh},$$

where ν_j is the number of system reliability posterior draws in bin B_j.

Next, each candidate allocation is evaluated in turn, using the following procedure to compute the expected entropy $\tilde{G}(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2)$. First, the number of outcomes possible for the given candidate allocation is determined as

$$P = \left(\prod_{i=1}^{k} (n_{2,i} + 1) \right) (n_{2,\theta} + 1).$$

All outcomes are considered sequentially, starting with the outcome in which all tests fail, $(0, 0, \ldots, 0)$. At no time are all outcomes stored in memory simultaneously, which greatly reduces the amount of memory used during computations.

For a specific outcome $\mathbf{x}_2$, $\tilde{p}(\mathbf{x}_2|\theta)$ and $\tilde{f}(\theta|\mathbf{x}_1, \mathbf{x}_2)$ are also represented as vectors of length n_h. We initialize the value of $\tilde{p}(\mathbf{x}_2|\mathbf{x}_1)$ to be 0. We fill in the entries of $\tilde{p}(\mathbf{x}_2|\theta)$ bin by bin, and simultaneously determine each bin's contribution to $\tilde{p}(\mathbf{x}_2|\mathbf{x}_1)$. The j^{th} entry of $\tilde{p}(\mathbf{x}_2|\theta)$, denoted $\tilde{p}_j(\mathbf{x}_2|\theta)$, is computed as

$$\tilde{p}_j(\mathbf{x}_2|\theta) = \frac{\sum_{l=1}^{M} p(\mathbf{x}_2|\mathbf{n}_2, p_1^{(l)}, p_2^{(l)}, \ldots, p_k^{(l)}, \theta^{(l)}) I(\theta^{(l)} \in B_j)}{\nu_j},$$

where $p(\mathbf{x}_2|\mathbf{n}_2, p_1^{(l)}, p_2^{(l)}, \ldots, p_k^{(l)}, \theta^{(l)})$ is defined in Equation 2.6, and the contribution of bin B_j to $\tilde{p}(\mathbf{x}_2|\mathbf{x}_1)$ is

$$\frac{\tilde{p}_j(\mathbf{x}_2|\theta)\nu_j}{M}, \qquad \text{for } \theta \in B_j.$$

After all bins have been considered, we will have a vector containing $\tilde{p}(\mathbf{x}_2|\theta)$ and

$$\begin{aligned}\tilde{p}(\mathbf{x}_2|\mathbf{x}_1) &= \sum_{j=1}^{n_h} \frac{\tilde{p}_j(\mathbf{x}_2|\theta)\nu_j}{M} \\ &= \sum_{j=1}^{n_h} \frac{\frac{\sum_{l=1}^{M} p(\mathbf{x}_2|\mathbf{n}_2, p_1^{(l)}, p_2^{(l)}, \ldots, p_k^{(l)}, \theta^{(l)}) I(\theta^{(l)} \in B_j)}{\nu_j}\nu_j}{M} \\ &= \sum_{l=1}^{M} \frac{p(\mathbf{x}_2|\mathbf{n}_2, p_1^{(l)}, p_2^{(l)}, \ldots, p_k^{(l)}, \theta^{(l)})}{M}.\end{aligned}$$

Now the three factors can be used to compute the updated system reliability posterior distribution. The j^{th} entry of the vector representing the updated posterior distribution, denoted $\tilde{f}_j(\theta|\mathbf{x}_1, \mathbf{x}_2)$, is

$$\tilde{f}_j(\theta|\mathbf{x}_1, \mathbf{x}_2) = \frac{\tilde{f}_j(\theta|\mathbf{x}_1)\tilde{p}_j(\mathbf{x}_2|\theta)}{\tilde{p}(\mathbf{x}_2|\mathbf{x}_1)}, \qquad \text{for } \theta \in B_j.$$

The entropy associated with this given outcome is then computed as

$$\tilde{H}(\theta|\mathbf{x}_1, \mathbf{x}_2) = -\sum_{j=1}^{n_h} h\tilde{f}_j(\theta|\mathbf{x}_1, \mathbf{x}_2) \log\left(\tilde{f}_j(\theta|\mathbf{x}_1, \mathbf{x}_2)\right),$$

where 0 log 0 is defined to be 0.

The entropy associated with the outcome $\mathbf{x}_2$ is used to find that outcome's contribution to the expected entropy ($\tilde{G}(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2)$)

$$\tilde{p}(\mathbf{x}_2|\mathbf{x}_1)\tilde{H}(\theta|\mathbf{x}_1, \mathbf{x}_2).$$

The remaining outcomes for the allocation are considered as above, and after $\tilde{H}(\theta|\mathbf{x}_1, \mathbf{x}_2)$ has been computed for all $\mathbf{x}_2$, the expected entropy for that allocation is computed as

$$\tilde{G}(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2) = \sum_{\mathbf{x}_2} \tilde{p}(\mathbf{x}_2|\mathbf{x}_1)\tilde{H}(\theta|\mathbf{x}_1, \mathbf{x}_2).$$

The process is then repeated for each candidate allocation.

2.4.1 Example: Evaluating Candidates for a Five Component Series System

As an example of how this new approach for evaluating the planning criterion associated with a candidate allocation can be implemented, recall the five-component series system in

Figure 1.4. Suppose that first-stage data are available for each component in the system and that a single expert specifies a best guess about the reliability of each component. The first-stage data, expert guesses and testing costs are displayed in Table 2.4; the system tests are most expensive and the combined cost of one test of each basic component and the subsystem is the same as the cost as a single system test.

Table 2.4 First-stage data, expert best guesses, and tests costs for five component series system

Type	Component	Successes	Tests	π	Testing Cost
Basic	1	197	200	.99	\$1
Basic	2	199	200	.99	\$1
Basic	3	192	200	.95	\$1
Subsystem	4	99	100	.9801	\$2
System	5	38	40	.9311	\$5

The system reliability posterior distribution given the first-stage data, based on 10,000 posterior draws using the model from Section 1.2, is displayed in Figure 2.2. The system reliability posterior mean is 0.9416835 with posterior variance 0.0001926 and entropy - 2.85311. The central 95% credible interval for system reliability is (0.9119318, 0.9655372); the 90% credible interval is (0.9170802, 0.9625527). We now assume that there are resources available to collect second-stage data; the purpose of collecting this additional data is to obtain a more precise estimate of the system's reliability. We expect that the addition of more data will result in smaller variance and entropy, and narrower credible intervals.

Suppose that the budget for collecting second-stage data is \$100. There are many ways to allocate our resources among component tests. A possible allocation is (0, 0, 100, 0, 0), the allocation that uses all additional resources on Component 3 tests. This allocation is intuitive in that Component 3 had the worst performance according to the first-stage data. Another possible allocation would be to spend the entire budget on system tests, (0, 0, 0, 0, 20); we want to improve the precision associated with our estimate of system reliability, and thus performing as many full system tests as possible is a natural choice. Alternatively, the budget could be split equally, based on proportion of the budget, among all system components (20, 20, 20, 10, 4) or spent entirely on basic components (33, 33, 34, 0, 0). Another possibility would be

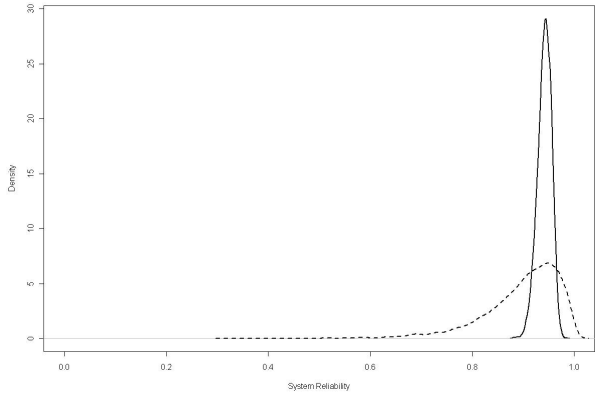


Figure 2.2 System reliability prior (dashed) and posterior (solid) distributions using the model described by Johnson *et al* (2003)

to split the budget between the subsystem (Component 4) and Component 3, resulting in the allocation (0, 0, 50, 25, 0).

Table 2.5 Expected entropy for the five candidate allocations

Allocation	Expected Entropy
(0, 0, 100, 0, 0)	-2.96517
(0, 0, 0, 0, 20)	-2.88536
(33, 33, 34, 0, 0)	-2.90815
(0, 0, 50, 25, 0)	-2.92426
(20, 20, 20, 10, 4)	-2.8957

The expected entropy for each of the five candidates, summarized in Table 2.5, was computed with a single call of the candidate evaluation program and is based on the output from the initial analysis via MCMC. The first four candidates displayed in Table 2.5 were evaluated fairly quickly (about an hour), while the last candidate took considerably longer (over 12 hours); in this dissertation all reported computing times are relative to an Intel® Xeon® 3.16 GHz processor, unless otherwise noted. Issues pertaining to evaluating large candidate allocations are addressed in Section 2.6. Of the candidate allocations considered, the allocation using all resources on Component 3 results in the lowest expected entropy and thus was the most informative allocation, while spending the entire budget on the full system was the least informative allocation, possibly because of the small number of full system tests allowed.

Although the allocations chosen in this example had some intuitive appeal, they are only a tiny fraction of the allocations that could be considered and there is no guarantee that one of these five is the overall best way to allocate the available resources. In Chapter 3, we describe how to combine the candidate evaluation methodology of this chapter with an optimization procedure for use when the number of possible allocations is large.

2.5 Numerical Error in the Histogram-Based Calculation of Entropy for Beta Distributions

An important issue is the amount of numerical error in the entropy calculation induced by the histogram approximation to the posterior density of θ, the system reliability. We address this issue in the context of a Beta distribution. While the posterior distribution of θ is unlikely to be a Beta distribution, these calculations provide some insight into choosing the bin width h when the calculating expected entropies of candidate allocations using the approach described in Section 2.3. The true entropy of the Beta distribution with parameters α and β is given by

$$H = \ln B(\alpha,\beta) - (\alpha-1)\left[\psi(\alpha) - \psi(\alpha+\beta)\right] - (\beta-1)\left[\psi(\beta) - \psi(\alpha+\beta)\right], \quad (2.13)$$

where $B(\alpha,\beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$ and $\psi(\cdot)$ is the digamma function, $\psi(z) = \frac{d}{dz}\Gamma(z)$ (Cover and Thomas (1991)), and the distributions Beta(α, β) and Beta(β, α) have the same entropy.

Next we define the *best fitting histogram* with bin width h as having bin height

$$p_j = \frac{1}{h}\int_{(j-1)h}^{jh} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}\,dx \qquad \text{for } x \in ((j-1)h, jh)\,;$$

that is, the area of Bin j with endpoints $(j-1)h$ and jh is equal to the integral of the Beta density curve between $(j-1)h$ and jh. Suppose that the bin width h is chosen so that $h = \sigma$, where σ is the standard deviation of the Beta distribution, that is:

$$\sigma = \sqrt{\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}}.$$

The number of bins used to construct the best fitting histogram with bin width h is $n_h = \lceil \frac{1}{h} \rceil$; the ceiling is used to ensure that the entire interval (0, 1) is covered by the histogram. To ensure that the distributions Beta(α, β) and Beta(β, α) have the same calculated entropy, the

construction of the histogram bins begins at 0 if the mean is less than (or equal to) 0.5 and at 1 if the mean is greater than 0.5.

Using the best fitting histogram to calculate the entropy of the Beta distribution yields

$$H^* = -\sum_j hp_j \log(p_j). \tag{2.14}$$

The amount of numerical error in the entropy calculation is

$$\xi = H^* - H,$$

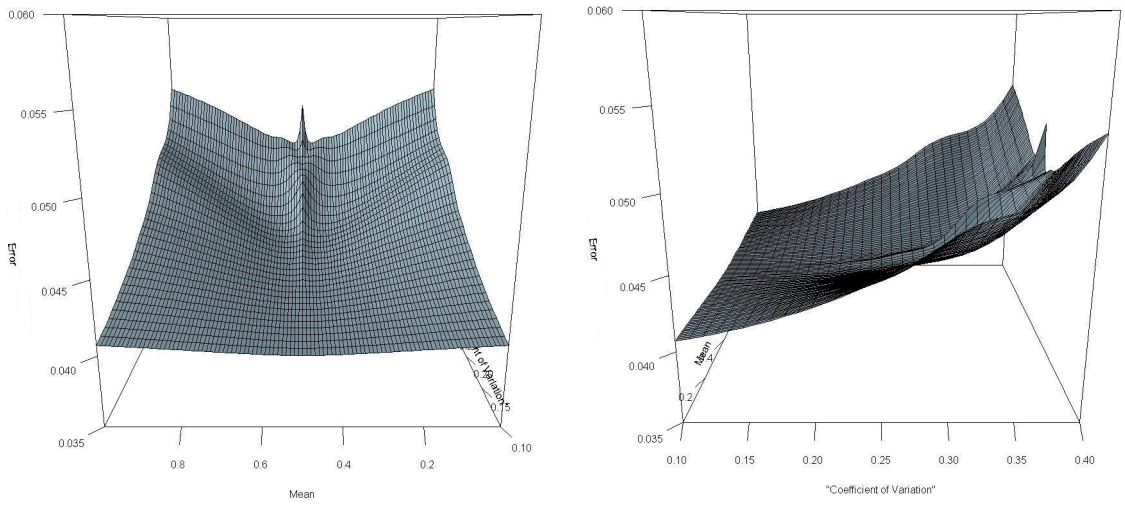
where H is the true entropy computed using Equation 2.13 and H^* is the histogram-based entropy from Equation 2.14. The amount of numerical error depends on both the mean and standard deviation of the Beta distribution; two Beta distributions with the same mean do not necessarily have the same entropy and neither do two Beta distributions with the same standard deviation. We define

$$c_v = \begin{cases} \frac{\sigma}{\mu} & \text{if } \mu \leq 0.5 \\ \frac{\sigma}{1-\mu} & \text{if } \mu > 0.5 \end{cases}$$

For $\mu \leq 0.5$, this is simply the coefficient of variation. Note that the distributions Beta(α, β) and Beta(β, α) both have standard deviation σ; if Beta(α, β) has mean μ, then Beta(β, α) has mean $1-\mu$. Since these two distributions have the same entropy, they have the same error when the entropy is computed using a histogram.

Figures 2.3(a) and 2.3(b) display the relationship between the amount of numerical error, the mean of the distribution ($0.01 \leq \mu \leq 0.99$), and c_v ($0.1 \leq c_v \leq 0.4$). For $c_v > 0.4$ (not displayed here) the amount of numerical error increases drastically, especially for extreme means ($\mu \geq 0.8$ and $\mu \leq 0.2$); when $c_v < 0.1$ and the mean is extreme, the true entropy cannot be evaluated because of underflow in the calculation of the Beta function.

One obvious feature of the relationship is that, for a given mean, the amount of numerical error decreases as c_v, and thus σ, decreases. For $h = \sigma$, the bin width decreases with σ and the number of bins used to construct the histogram increases. When the bin width is $h = \sigma$, an empirical lower bound on the numerical error seems to be about 0.04; for $c_v \leq 0.4$ an approximate upper bound might be about 0.056.



(a) (b)

Figure 2.3 Numerical error in the histogram-based entropy calculation for Beta distributions with specified mean and c_v

The greatest amount of numerical error occurs for the extreme means ($\mu \geq 0.8$ and $\mu \leq 0.2$) with high c_v. For our application, an extreme mean with large c_v corresponds to systems that are very reliable (or unreliable), but under conditions where very little information is available about the system's reliability (i.e., a relatively large variance). There is also a peak at $\mu = 0.5$ when $c_v = 0.4$; this combination of μ and c_v implies $\sigma = 0.4 \times 0.5 = 0.2$ which corresponds to Beta(2.625, 2.625). A bin width of $h = 0.2$ implies 6 bins are used, which yields a poor histogram approximation to the Beta(2.625, 2.625) density and a rather poor estimate of the entropy.

Figure 2.6 displays the numerical error for four more bin widths, each chosen to be a fraction of σ ($h = \frac{\sigma}{2}$, $h = \frac{\sigma}{4}$, $h = \frac{\sigma}{6}$, $h = \frac{\sigma}{10}$). For each bin width, the overall shape of the error surface remains the same; extreme means and $\mu = 0.5$ with high c_v have the most numerical error due to the histogram calculation of entropy, with the amount of numerical error decreasing with c_v. When $h = \frac{\sigma}{2}$ (Figure 2.4(a)), the numerical error tends to be between 0.01 and 0.017. Approximate bounds on the numerical error when $h = \frac{\sigma}{4}$ are 0.0025 and 0.0045. Bounds on the amount of numerical error for $h = \frac{\sigma}{6}$ and $h = \frac{\sigma}{10}$ are roughly 0.0012 to 0.0022 and 0.0004 to 0.0008, respectively.

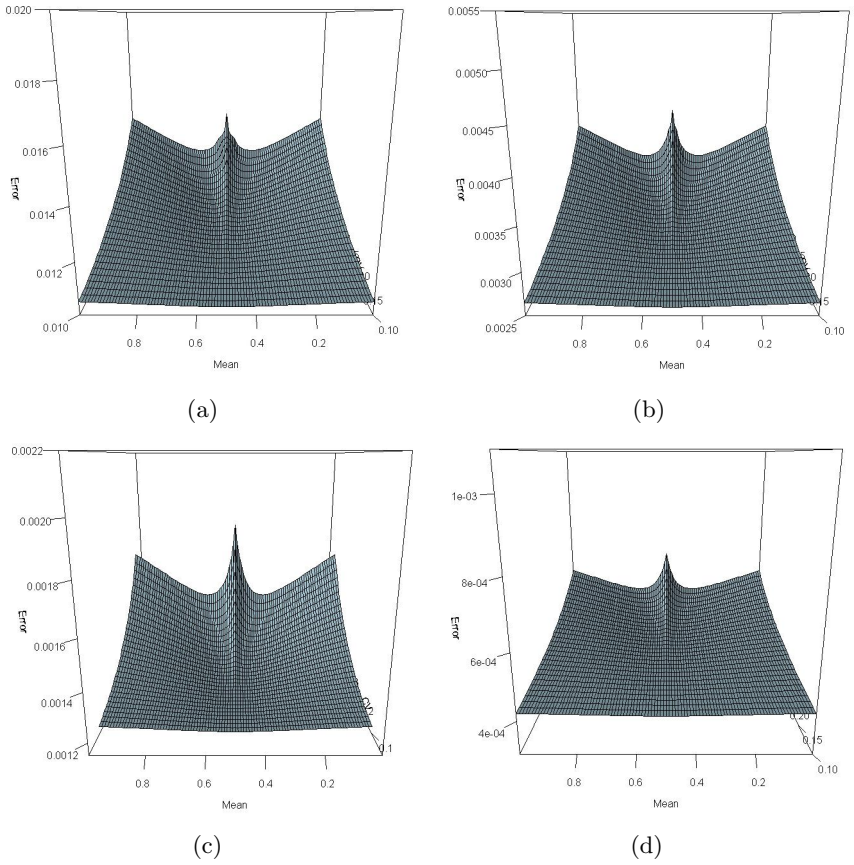


Figure 2.4 Numerial error in the histogram-based entropy calculation for bin widths (a) $h = \frac{\sigma}{2}$, (b) $h = \frac{\sigma}{4}$, (c) $h = \frac{\sigma}{6}$, and (d) $h = \frac{\sigma}{10}$

In Table 2.6 we consider the amount of numerical error in the histogram-based entropy calculation with bin widths $h = \sigma$, $h = \frac{\sigma}{3}$, and $h = \frac{\sigma}{10}$, for Beta distributions with various means and standard deviations; the means chosen are intended to reflect a sensible range for system reliabilities. We see that for $\sigma = \frac{(1-\mu)}{5}$ and $\sigma = \frac{(1-\mu)}{10}$, the numerical error is consistent across the range of means considered, and decreases as the bin width decreases; both $\sigma = \frac{(1-\mu)}{5}$ and $\sigma = \frac{(1-\mu)}{10}$ are cases in which $c_v < 0.4$. Based on this table we would expect the numerical error in using a histogram to calculate the entropy of a Beta distribution with mean 0.77 and standard deviation 0.046 ($\sigma = \frac{1-\mu}{5}$) using $h = \sigma$ to be between 0.041417 and 0.041747; the numerical error is actually 0.041549. For $\sigma = \frac{(1-\mu)}{5}$ and $\sigma = \frac{(1-\mu)}{10}$ the numerical error is positive, indicating that the calculated entropy ($\tilde{H}$) is larger than the true entropy (H).

For $\sigma = (1 - \mu)$ and $\sigma = \frac{(1-\mu)}{2}$, the magnitude of the numerical error tends to be larger

Table 2.6 Numerical error in the histogram-based entropy calculation using bin width h for various Beta distributions

		$\mu = .7$	$\mu = .75$	$\mu = .8$	$\mu = .85$	$\mu = .9$	$\mu = .95$
$\sigma = 1 - \mu$	$h = \sigma$	0.095435	-0.087127	-0.15932	-0.20271	-0.22750	-0.24164
	$h = \frac{\sigma}{3}$	-0.14238	-0.25050	-0.30565	-0.33374	-0.34714	-0.35233
	$h = \frac{\sigma}{10}$	-0.19593	-0.25919	-0.27360	-0.26921	-0.25653	-0.24047
$\sigma = \frac{(1-\mu)}{2}$	$h = \sigma$	-0.24608	-0.23749	-0.22949	-0.22213	-0.21539	-0.20922
	$h = \frac{\sigma}{3}$	-0.056540	-0.046667	-0.038476	-0.031650	-0.025931	-0.021115
	$h = \frac{\sigma}{10}$	-0.005255	-0.003213	-0.001874	-0.000990	-0.00040	-1.955×10^{-6}
$\sigma = \frac{(1-\mu)}{5}$	$h = \sigma$	0.041088	0.041417	0.041747	0.042070	0.042383	0.042682
	$h = \frac{\sigma}{3}$	0.004755	0.004797	0.004840	0.004883	0.004924	0.004963
	$h = \frac{\sigma}{10}$	4.3×10^{-4}	4.34×10^{-4}	4.38×10^{-4}	4.42×10^{-4}	4.46×10^{-4}	4.49×10^{-4}
$\sigma = \frac{(1-\mu)}{10}$	$h = \sigma$	0.04025	0.04034	0.04042	0.04050	0.04058	0.04066
	$h = \frac{\sigma}{3}$	0.004639	0.004650	0.004661	0.004671	0.004682	0.004691
	$h = \frac{\sigma}{10}$	4.19×10^{-4}	4.20×10^{-4}	4.21×10^{-4}	4.22×10^{-4}	4.23×10^{-4}	4.24×10^{-4}

and is much more variable as the mean changes than in the other two cases. In these cases, the numerical error tends to be, though is not always, negative, indicating a tendency of the calculated entropy to be less than the true entropy. The large numerical error, the sign of the numerical error, and/or variability in the numerical error could be due to the placement of the histogram bins relative to the Beta density curve. In this investigation, the construction of the bins begins at either 0 or 1 (depending on whether the mean is above or below 0.5); the amount of error might be reduced by considering a subset of the interval (0, 1) that is more appropriate for the specific Beta distribution.

In reality, we are not assuming that the system reliability posterior distribution given the first-stage data is a Beta distribution; however, it is quite reasonable to believe that the system reliability posterior distribution given the initial data is somewhat close to a Beta distribution. We can get an idea of roughly how much numerical error there is in our histogram-based calculation of entropy for the posterior distribution by computing the "true entropy" to be the entropy of a Beta distribution with the same mean and standard deviation as the system reliability posterior distribution; the difference between the "true entropy" and the calculated entropy can be used as a rough assessment of the amount of numerical error.

This rough assessment of the amount of numerical error can then be used to gauge how

much error there might be in the computation of the expected entropy of an allocation. To see this, first note that the goal of the resource allocation study is to obtain a more precise estimate of the system's reliability. We typically do not expect a point estimate of system reliability to change much with the addition of more data. We are, however, expecting to see a reduction in the amount of uncertainty there is in the posterior distribution. Thus, the updated system reliability posterior distribution for most of the outcomes considered while computing the expected entropy for a candidate allocation should have a c_v value similar to, or less than, that of the posterior distribution given the first-stage data; thus the numerical error incurred when calculating the expected entropy for an allocation might be roughly that observed when calculating the entropy of the system reliability posterior distribution given the first-stage data. This provides us with an approximate guideline for comparing the expected entropy of candidate allocations; if the difference in the expected entropy of two candidate allocations is less than amount of error, we cannot conclusively say that one allocation is more informative than the other. Further, due to the small observed differences in the calculated entropies for candidate allocations (e.g., the example in Section 2.4.1), this investigation suggests that a bin width that is a small fraction of the standard deviation is necessary to ensure that candidates can be clearly separated.

In the example from Section 2.4.1, the bin width was initially chosen to be 0.01; this is approximately $h = \frac{\hat{\sigma}}{1.39}$. If we assume that the posterior distribution of system reliability can be approximated by a Beta distribution with $\mu = 0.9416835$ and $\sigma = 0.01387804$, the corresponding parameters are $\alpha = 267.5588$ and $\beta = 16.5694$. The true entropy for Beta(267.5588, 16.5694) is -2.875341; using bin width $h = 0.01$, the entropy was computed to be -2.875488. Thus we may expect the error to be roughly 0.022378, or about $\xi = 0.0224$. If the system reliability posterior distribution truly was Beta(267.5588, 16.5694), the entropy calculated using the best fitting histogram with bin width $h = 0.01$ would be -2.852125. In this example, the most informative allocation was found to have expected entropy -2.96517. If we assume that the numerical error is roughly $\xi = 0.0224$, we might believe that the expected entropy for this allocation is actually somewhere between -2.9741 and -2.9293. Similarly, we might believe the

expected entropy of the allocation (0, 0, 50, 25, 0) to be between -2.94666 and -2.90186. These error bands for the top two candidate allocations overlap; thus due to the potential numerical error, we cannot conclude which allocation is actually better. The expected entropy for all five candidate allocations and approximate error bands for the expected entropy are displayed in Table 2.7.

Table 2.7 Expected entropy, computed using 100 bins, for the five candidate allocations and approximate error bands

Allocation	Expected Entropy	Expected Entropy - ξ	Expected Entropy + ξ
(0, 0, 100, 0, 0)	-2.9517	-2.9741	-2.9293
(0, 0, 0, 0, 20)	-2.88537	-2.90777	-2.86297
(33, 33, 34, 0, 0)	-2.90815	-2.93055	-2.88575
(0, 0, 50, 25, 0)	-2.92426	-2.94666	-2.90186
(20, 20, 20, 10, 4)	-2.8957	-2.9181	-2.8733

To reduce the amount of numerical error, these five candidate allocations were re-evaluated using a histogram with 721 bins; the bin width was roughly $\frac{\hat{\sigma}}{10}$. Now, the entropy of the system reliability posterior distribution was computed to be -2.87919; thus we may expect the amount of error to be roughly -0.004. If the best fitting histogram with bin width $h \approx \frac{\hat{\sigma}}{10}$ is used to compute the entropy of the Beta(267.5588, 16.5694) distribution, the calculated entropy would be -2.875023. The expected entropy for each candidate, as well as an error band for the expected entropy, is displayed in Table 2.8.

Table 2.8 Expected entropy, computed using 721 bins, for the five candidate allocations and approximate error bands

Allocation	Expected Entropy	Expected Entropy - ξ	Expected Entropy + ξ
(0, 0, 100, 0, 0)	-2.9966	-3.0006	-2.9926
(0, 0, 0, 0, 20)	-2.91291	-2.91691	-2.90891
(33, 33, 34, 0, 0)	-2.93737	-2.94137	-2.93337
(0, 0, 50, 25, 0)	-2.95378	-2.95778	-2.94978
(20, 20, 20, 10, 4)	-2.92417	-2.92817	-2.92017

When the smaller bin width is used, the error band for the top candidate no longer overlaps with that for the second candidate; now we can feel confident that the top-ranked allocation truly is more informative than the rest of the allocations considered. Further, there is no overlap

in the error bands for any of these allocations, and thus we can clearly rank the allocations in order of their informativeness. We notice that the bands constructed in Table 2.8 for a given allocation do not overlap with those for the same allocation in Table 2.7, even though we expect them to overlap; the fact that they do not could be due to the fact that we aren't actually working with a Beta distribution or a result of compounded rounding error in the calculations. More important is the fact that the relative ranking of the candidate allocations did not change when more bins were used to compute the expected entropy; this suggests, though does not prove, that the error in the difference between the calculated candidate expected entropies is less than the error the numerical error associated with the calculation of individual candidate allocations.

2.6 Evaluation of Large Candidate Allocations

With the proposed approach, any number of candidates can, in principle, be evaluated using the output from the initial analysis. However, to evaluate the expected entropy for a candidate allocation, the proposed approach requires the enumeration of all possible outcomes for that candidate. If the candidate allocation is $\mathbf{n}_2 = (n_{2,1}, n_{2,2}, \ldots, n_{2,k}, n_{2,\theta})$, the number of possible outcomes P is

$$P = \left(\prod_{i=1}^{k} (n_{2,i} + 1) \right) (n_{2,\theta} + 1) .$$

The number of possible outcomes P is reasonable for small systems, but increases quickly as either the number of tests performed on components or the number of total components in the system increases. As the size of the problem grows, the feasibility of this complete enumeration of outcomes drastically decreases. Table 2.9 helps to illustrate the magnitude of this increase in the number of possible outcomes. In this illustration, the systems have 4-10 total components (including the full system) and the number of tests to be performed ranges from 5 to 100 tests per component; we are not concerned with component costs or adhering to a budget in this illustration. It is clear that increasing either the size of the experiment or the size of the system increases the possible number of outcomes, but system size is the more critical dimension as

increasing the size of the system by one component increases the number of outcomes more quickly than increasing component test sizes by one component.

Table 2.9 indicates how quickly a problem can grow, even for systems of moderate size. In many applications, typical systems considered may include substantially more components; test sizes for real systems can reach several hundred tests per component, with system tests being a little lower in number. Even the largest problem considered in Table 2.9 is potentially smaller than some real problems that may be encountered.

To evaluate large candidate allocations with the computational efficiency of the methodology in Section 2.3, we propose a modification in which we draw a sample of N possible outcomes $\left\{\mathbf{x}_2^{(1)}, \mathbf{x}_2^{(2)}, \ldots, \mathbf{x}_2^{(N)}\right\}$ for the candidate allocation $\mathbf{n}_2$ and only compute the entropy of the updated system reliability posterior distribution,

$$\tilde{H}(\theta|\mathbf{x}_1, \mathbf{x}_2) = -\sum_{\theta} \tilde{p}(\theta|\mathbf{x}_1, \mathbf{x}_2) \log\left(\tilde{p}(\theta|\mathbf{x}_1, \mathbf{x}_2)\right),$$

for each of these outcomes rather than for all possible outcomes, as in Section 2.3. The average of these N entropies can then be used to estimate the expected entropy for the candidate allocation. By using this sampling-based modification, we only need to evaluate a (comparatively) small number of outcomes, still using only a single MCMC analysis of the first-stage data, to estimate the expected entropy $G(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2)$.

On the surface, this sampling-based modification may sound similar to the current approach for evaluating the planning criterion associated with a candidate allocation. The distinction is that the current approach samples outcomes from a candidate allocation and, for each outcome, requires a new analysis be performed via MCMC, while our sampling-based approach uses the results from the initial MCMC analysis to compute the updated posterior entropy associated with each of the sampled outcomes.

To see this in more detail, consider a candidate allocation $\mathbf{n}_2$. To sample a single outcome from $p(\mathbf{x}_2|\mathbf{x}_1)$, we first draw an vector of component reliabilities, (p_1, ..., p_k, θ), from the joint posterior distribution. Next, for each component in the candidate allocation $\mathbf{n}_2 = (n_{2,1}, n_{2,2}, \ldots, n_{2,k}, n_{2,\theta})$, an outcome $x_{2,i}$ is drawn from the Binomial distribution with size $n_{2,i}$ and probability p_i, $i = 1, 2, \ldots, k, \theta$; the sampled outcome is $\mathbf{x}_2 = (x_{2,1}, x_{2,2}, \ldots, x_{2,k}, x_{2,\theta})$.

Table 2.9 Effect of increasing system size and experiment size on the number of possible outcomes

		Total Number of Components						
		4	5	6	7	8	9	10
Tests Per Component	5	1296	7776	46656	279936	1679616	10077696	60466176
	6	2401	16807	117649	823543	5764801	40353607	282475249
	7	4096	32768	262144	2097152	16777216	134217728	1.073741×10^9
	8	6561	59049	531441	4782969	43046721	387420489	3.286784×10^9
	9	10000	100000	1000000	10000000	100000000	1.0×10^9	1.0×10^{10}
	10	14641	161051	1771561	19487171	214358881	2.357947×10^9	2.593742×10^{10}
	25	456976	11881376	308915776	8.031810×10^9	2.088270×10^{11}	5.429504×10^{12}	1.411671×10^{14}
	50	6765201	345025251	1.759629×10^{10}	8.974107×10^{11}	4.576794×10^{13}	2.334165×10^{15}	1.190424×10^{17}
	100	1.040604×10^8	1.051010×10^{10}	1.061520×10^{12}	1.072135×10^{14}	1.082857×10^{16}	1.093685×10^{18}	1.104622×10^{20}

This process is repeated until a sample of size N outcomes has been drawn from $p(\mathbf{x}_2|\mathbf{x}_1)$. Let $\mathbf{x}_2^{(j)}$ denote the j^{th} sampled outcome. To compute the entropy of the updated system reliability posterior distribution given the first-stage data and the hypothetical outcome $\mathbf{x}_2^{(j)}$, recall Equation 2.10:

$$\tilde{f}(\theta|\mathbf{x}_1, \mathbf{x}_2) = \frac{\tilde{p}(\mathbf{x}_2|\theta)\tilde{f}(\theta|\mathbf{x}_1)}{\tilde{p}(\mathbf{x}_2|\mathbf{x}_1)}.$$

The three factors in the updated system reliability posterior distribution (Equation 2.10) can be computed, for $\mathbf{x}_2^{(j)}$, as detailed in Equations 2.7, 2.8, and 2.9. Then, for $\mathbf{x}_2^{(j)}$, the entropy of the updated system reliability posterior distribution can be computed as

$$\tilde{H}(\theta|\mathbf{x}_1, \mathbf{x}_2^{(j)}) = -\sum_{\theta} \tilde{p}(\theta|\mathbf{x}_1, \mathbf{x}_2^{(j)}) \log\left(\tilde{p}(\theta|\mathbf{x}_1, \mathbf{x}_2^{(j)})\right), \qquad j = 1, \ldots, N$$

We use

$$\hat{G}(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2) = \frac{1}{N}\sum_{j=1}^{N} \tilde{H}(\theta|\mathbf{x}_1, \mathbf{x}_2^{(j)})$$

to estimate $\tilde{G}(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2)$; because the outcomes $\mathbf{x}_2$ are sampled from $p(\mathbf{x}_2|\mathbf{x}_1)$ and

$$\tilde{G}(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2) = \sum_{\mathbf{x}_2} \tilde{p}(\mathbf{x}_2|\mathbf{x}_1)\tilde{H}(\theta|\mathbf{x}_1, \mathbf{x}_2),$$

the Law of Large Numbers ensures that $\hat{G}(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2)$ converges with probability one to $\tilde{G}(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2)$ as N approaches ∞ (Casella and Berger (2002)).

2.6.1 Demonstration: Evaluating Large Candidate Evaluations Efficiently

As a demonstration of how this sampling-based modification for large allocations can be used, recall the example from Section 2.4.1. The five allocations were evaluated using the sampling modification with five different sample sizes (N = 100, 500, 1000, 5000, and 10,000) and 721 bins. For each candidate allocation, the expected entropy ($\tilde{G}(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2)$ computed using the methodology in Section 2.3) and the estimated expected entropy ($\hat{G}(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2)$), standard deviation of the sampled entropies ($s_{\hat{G}}$), and a 95% normal-theory confidence interval for the expected entropy, to account for the variability due to sampling outcomes, for each sample size are displayed in Table 2.10.

First consider evaluating the five candidate allocations using a sample of size 100 for each candidate. The allocation (0, 0, 100, 0, 0), which was considered to be the best allocation using the evaluation methodology from Section 2.3, has the highest estimated entropy, and the 95% confidence interval for the expected entropy does not overlap with those for the other allocations. Thus, we can conclude that this allocation is the most informative of the five allocations. The allocation that uses the entire budget on system tests, (0, 0, 0, 0, 20), has the lowest estimated expected entropy and the corresponding confidence interval does not overlap with those for any other allocation, leading us to believe that this allocation is the least informative of those considered. The confidence intervals for the three remaining allocations overlap, and thus we can not rank these allocations according to their expected informativeness.

For all sample sizes considered, the allocation using all resources on Component 3 is clearly the most informative allocation, while the allocation utilizing the entire budget on system tests was clearly the least informative. For samples of size 500 or more, the allocation that divides the resources evenly between Components 3 and 4 is clearly the second most informative allocation, as the associated confidence intervals do not overlap with the confidence intervals for the expected entropy of any other allocation. It is not clear as to how the two remaining allocations should be ranked until samples of size 5000 and 10,000 are used. When samples of size 5000 and 10,000 are used, the allocation dividing all resources evenly between the basic components is clearly the third most informative allocation, though it does not seem to provide much more information than the fourth most informative allocation.

For each allocation, the 95% confidence intervals for the expected entropy, based on the five different sample sizes, contained the expected entropy ($\tilde{G}$), suggesting that this sampling-based candidate evaluation technique is a sensible way to estimate the expected entropy for a candidate allocation. Of the sample sizes considered, evaluating candidates based on samples of size 10,000 is the most time-consuming, generally requiring several minutes to evaluate a single candidate allocation while only a few seconds are required to evaluate a single candidate allocation based on a sample of size 100. When the goal is to evaluate a few candidate allocations precisely, using samples of size 10,000 is a reasonable choice and is less computationally

taxing than evaluating candidates by fully enumerating all possible outcomes. However, the choice of sample size ultimately depends on the number of candidates to be evaluated, the available computing time, and how precisely candidates need to be evaluated.

2.7 Chapter Summary

In this chapter we discuss various aspects of evaluating the amount of information to be expected from performing a candidate experiment. First we address the issue of selecting a planning criterion. Throughout this chapter we promote the use of posterior entropy as a planning criterion. Under the Bayesian experimental design framework, finding the allocation that minimizes the expected entropy of the updated system reliability posterior distribution is equivalent to finding the allocation that maximizes the expected gain in Shannon information.

Computationally efficient candidate evaluation methodology is also introduced in this chapter. The approach currently used to evaluate candidate allocations is computationally expensive and requires multiple runs of an MCMC algorithm. Our alternative methodology allows multiple candidate allocations to be evaluated with only a single MCMC analysis. The methodology described in Section 2.3 is typically only feasible for problems of limited size, as it requires all possible outcomes from a candidate allocation to be enumerated. In Section 2.6, we provide a sampling-based modification of this methodology that allows us to evaluate candidates from any allocation, while only using a single MCMC analysis.

All of the methodology introduced in this chapter relies on a histogram approximation to a continuous distribution. Inevitably, there is going to be some amount of numerical error associated with this use of a histogram to compute the entropy of the distribution. We investigate the behavior of this error in the case of a Beta distribution. In practice, we will not necessarily be working with Beta distributions, but the posterior distributions in question will typically be somewhat similar to a Beta distribution and the insight gained from this investigation can be used to determine an appropriate bin width for the histogram. From this investigation we conclude that a bin width that is a small fraction of the posterior standard deviation, e.g., $h \approx \frac{\hat{\sigma}}{10}$, will typically result in a very small amount of numerical error in the histogram-based

entropy calculation.

This chapter contains numerous examples that demonstrate how this new, computationally efficient methodology can be used to evaluate a specified set of candidate allocations. The ultimate goal of resource allocation studies is to find an optimal, or nearly-optimal, allocation of the available resources; that is, an allocation that will give us as much new information as possible about the reliability of the system being studied. Because candidate allocations can be evaluated so efficiently with this new methodology, the methodology works well when paired with an optimization procedure. In Chapter 3 we describe how this new methodology and genetic algorithms can be used to search for an optimal resource allocation.

Table 2.10 The expected entropy compute for each candidate allocation and the estimate of expected entropy, standard deviation of the sampled entropies, and 95% confidence interval for the expected entropy, based on samples of size N=100, 500, 1000, 5000, and 10,000

Allocation	$\tilde{G}$	N	$\hat{G}$	$s_{\hat{G}}$	95% CI for $\tilde{G}$
(0, 0, 100, 0, 0)	-2.9966	100	-2.99377	0.0455268	(-3.002693, -2.984847)
		500	-2.99703	0.0457119	(-3.001037, -2.993023)
		1000	-2.99679	0.0472094	(-2.999716, -2.993864)
		5000	-2.99752	0.0458674	(-2.998791, -2.996249)
		10000	-2.99651	0.0457002	(-2.997406, -2.995614)
(0, 0, 50, 25, 0)	-2.95378	100	-2.94727	0.0351098	(-2.954152, -2.940388)
		500	-2.95689	0.0388209	(-2.960293, -2.953487)
		1000	-2.95137	0.0400479	(-2.953852, -2.948888)
		5000	-2.95309	0.0387168	(-2.954163, -2.952017)
		10000	-2.95342	0.0390929	(-2.955086, -2.953554)
(33, 33, 34, 0, 0)	-2.93737	100	-2.94511	0.0308709	(-2.951161, -2.939059)
		500	-2.93708	0.0338289	(-2.940045, -2.934115)
		1000	-2.9367	0.0346645	(-2.938849, -2.934551)
		5000	-2.93792	0.0344019	(-2.938874, -2.936966)
		10000	-2.93779	0.0337607	(-2.938452, -2.937128)
(20, 20, 20, 10, 4)	-2.92417	100	-2.92648	0.0294089	(-2.932244, -2.920716)
		500	-2.92323	0.0304182	(-2.925896, -2.920564)
		1000	-2.92489	0.02977912	(-2.926736, -2.923044)
		5000	-2.92373	0.0301667	(-2.924566, -2.922894)
		10000	-2.92465	0.0297721	(-2.925234, -2.924066)
(0, 0, 0, 0, 20)	-2.91291	100	-2.9158	0.0234219	(-2.920391, -2.911209)
		500	-2.91314	0.0268614	(-2.915495, -2.910785)
		1000	-2.91281	0.0266489	(-2.914462, -2.911158)
		5000	-2.91273	0.0271437	(-2.913482, -2.911978)
		10000	-2.9129	0.0267597	(-2.913424, -2.912376)

CHAPTER 3. GENETIC ALGORITHMS FOR OPTIMAL RESOURCE ALLOCATION STUDIES

In Chapter 2 we introduced a new approach for evaluating a candidate allocation which is much more computationally efficient than procedures that had been used previously (Hamada *et al* (2004), Wilson *et al* (2006), Hamada *et al* (2008)). The primary practical drawback to this new methodology is the requirement that all possible outcomes for a candidate allocation be enumerated, which is infeasible if the size of the problem is very large. A sampling-based modification of our approach was also presented, which allows large candidate allocations to be evaluated while still reducing the amount of computing power and time needed. However, evaluating a single candidate allocation is only a small part of the problem.

The goal of these resource allocation studies is to find an optimal (or at least very good) way to spend resources on a new experiment. In such problems, there may be many possible allocations that satisfy the budget constraints. Evaluating all of them may be impossible, but we still need to evaluate many allocations to determine which one we expect to be the most informative. Using previous methods, one of the greatest challenges faced in finding such an optimal allocation has been the computational burden (i.e., repeated MCMC calculations) necessary to evaluate a single allocation. The methodology introduced in Chapter 2 makes it much easier to evaluate candidate allocations using the results of a single analysis via MCMC. For that reason, the methods described in Chapter 2 can easily be combined with an optimization procedure to find an optimal allocation of resources. Hamada *et al* (2004) suggest the use of genetic algorithms to search for an optimal resource allocation and we shall use that approach here.

In this chapter, we give a brief introduction to genetic algorithms and how they can be used

in optimization problems. After the general description, we present an implementation of a genetic algorithm for finding an optimal allocation using the candidate evaluation procedures from Chapter 2 and provide several small examples. Larger examples of the use of genetic algorithms for finding optimal allocations will be presented in Chapter 4.

3.1 Introduction to Genetic Algorithms

A genetic algorithm (GA) is a numerical optimization technique that mimics the evolutionary process (Michalewicz (1996), Goldberg (1989), De Jong (2006)). GAs are often used as search algorithms or in optimization problems (De Jong (2006)). The process begins by creating an initial *population* of potential problem solutions. For illustration purposes, in this section we will represent a population member (solution) as a real-valued vector with five entries. Each entry of the vector can be thought of a *gene*; when an individual reproduces, it passes one or more genes to its offspring. This solution representation is known as *phenotypic* representation and is not the representation traditionally used to represent solutions in genetic algorithms (De Jong (2006)). *Genotypic* representation is the typical representation for candidate solutions in GAs and uses a binary vector, called a *chromosome*, to represent an individual in the population (Michalewicz (1996)); this chromosome can be decomposed into shorter segments that represent the different genes. Genotypic representation has historically been chosen to make the reproductive mechanisms independent of the application whereas phenotypic representation uses a more meaningful representation at the expense of making the reproductive mechanisms application-specific. For example, an integer-valued vector with five entries could meaningfully represent a solution when the application is to find an optimal allocation of resources for a five-component system.

Each member of the population has an inherent ability to survive, or *fitness*, which is being optimized by the GA. In the optimal resource allocation application, expected entropy is the fitness of a candidate solution and we use the GA to find the allocation with the lowest expected entropy. Then, as with a biological population, the members of this initial population of solutions must compete for the ability to survive and reproduce; ideally the stronger members

of the population (i.e., the better solutions) will be more likely to survive and reproduce than the weaker members (i.e., the worst solutions). When a candidate solution is selected to be a *parent* solution, it can produce *offspring* via either *mutation* (single parent reproduction, resulting in a *mutation* solution) or *recombination* (multiple parent reproduction, resulting in a *child* solution).

Once offspring have been produced, the algorithm can proceed in one of two ways. If the algorithm is *non-overlapping*, the new offspring will replace their parents to create a new generation, at the risk of the most fit member (best solution) dying off (De Jong (2006)). In the case of a search algorithm, this may mean moving away (at least temporarily) from the best solution seen so far to explore a different area of the search space. If the algorithm being used is *overlapping*, the new offspring will compete with the members of the parent population and only the top solutions will move on to the next generation (De Jong (2006)). In this instance, the strongest member of the population (best solution) in Generation g will be no weaker than the strongest member in Generation $g - 1$. This cycle of reproduction and survival is repeated until either a specified number of generations have been completed or some convergence criterion has been met. The following subsections provide some details about the mechanisms (selection and reproduction) in a genetic algorithm. More detail on genetic algorithms, as well as the broader field of evolutionary computation, can be found in Michalewicz (1996) and De Jong (2006).

3.1.1 Selection

Selection is the process by which population members (solutions) are chosen to produce offspring (Michalewicz (1996), De Jong (2006)). A variety of different selection strategies exist, with different strategies being appropriate for different problems. Each selection strategy has associated with it a degree of *selective pressure* which will impact the aggressiveness of the algorithm. When a strong selection strategy is employed, the most fit individuals in the population (i.e., those that have the lowest expected entropy) are more likely to be chosen to reproduce, resulting in a more homogeneous group of offspring solutions; a weaker selection

strategy tends to equalize the probability of being chosen to reproduce, thus creating a more diverse group of offspring solutions. In a search algorithm, strong selective pressure causes the algorithm to quickly focus on the first region or regions of the solution space found to have relatively good solutions, which allows for rapid convergence to the global optimum if it is in one of these regions, but may be problematic if these areas contain only local optima; weak selective pressure allows the algorithm to consider a wider variety of solutions, but generally take longer to find the optimal solution.

Selection methods can be either deterministic or stochastic. Deterministic strategies specify the exact number of times that each individual in the population produces offspring (De Jong (2006)). For example, suppose that there are m members of the population. One deterministic strategy may say that m offspring will be produced and that each member of the population will produce exactly one offspring. Another strategy may allow only for $n < m$ offspring and thus the n strongest members reproduce exactly once while the remaining members of the population do not produce any offspring. The latter deterministic selection strategy is known as truncation selection and is the strongest selection strategy (De Jong (2006)). Stochastic strategies randomly choose parents to produce offspring, with the selection probability depending on the selective pressure of the method; the variability introduced by the stochastic selection strategies can help prevent the algorithm from converging to a local optimum. De Jong (2006) illustrates the implications of using deterministic and stochastic selection strategies on the homogeneity of the populations in the genetic algorithm.

A few of the common stochastic selection strategies are described here, in order from weakest selective pressure to the strongest selective pressure; Michalewicz (1996) and De Jong (2006) provide more complete discussions of the various selection strategies. The weakest stochastic selection strategy is *uniform selection*. Under uniform selection, all individuals in the population have the same chance of reproducing, which will tend to result in a more heterogeneous group of offspring solutions. *Fitness-proportional* selection is another possible strategy. In this case, the fitness of each population member (solution) is determined, with f_i representing the fitness of the i^{th} individual and $\sum_i f_i$ denoting the overall fitness. Then, the probability of

selection for individual i is then $f_i / \sum_i f_i$; individuals that have the highest fitness relative to the overall fitness will be more likely to be chosen to reproduce. De Jong (2006) notes that fitness-proportional selection can be somewhat elitist (i.e., tends to favor the allocations with the best fitness) in early generations but tends to become more uniform in the later generations. A third selection strategy is *rank-proportional* selection; under this selection strategy, individuals with a higher rank are more likely to be chosen to produce offspring than individuals with a lower ranking. To illustrate the increase in selective pressure from fitness-proportional selection to rank-proportional selection, suppose that the two strongest individuals in a generation differ in their fitness only by a small amount ϵ; under fitness-proportional selection the probability that one of these two individuals is chosen to reproduce is quite similar, while under rank-proportional selection their selection probabilities will be weighted by their first and second rankings, regardless of the actual difference in their fitness values. De Jong (2006) provides a detailed discussion of the behavior of various selection techniques under different circumstances.

3.1.2 Reproduction

Reproduction is the means by which the search space is explored (De Jong (2006)). Once the candidates have been selected to reproduce, they can produce offspring via either mutation (single parent reproduction) or recombination (multiple parent reproduction). A mutation solution is generated by randomly changing one or more of the genes in a single parent solution to create a single offspring solution. When making a mutation solution, we are essentially taking a small step away from the parent solution in a randomly selected directed; this step size can be chosen by drawing an observation at random from some distribution. For example, suppose the parent allocation is (a, b, c, d, e) and a mutation of the first entry is chosen by randomly drawing a step size s from Normal$(0, \sigma)$; the result would be $(a + s, b, c, d, e)$ where the first gene is modified by the amount s. Note that σ can be fixed or vary according to the generation, and increasing σ increases the distance between a parent solution and its mutation solution. For example, Hamada *et al* (2004) allow the size of mutations to decrease in later

generations; that is, in early generations the mutation solutions are more likely to be further away from the parent solution than in the later generations. The mutation mechanism allows new genes to be introduced throughout the duration of the GA.

In recombination, two or more candidates are selected and mated to produce a child solution (Michalewicz (1996), De Jong (2006)); we will focus only on strategies involving two parents. Unlike mutation, when a recombination is made the resulting child solution contains only genes that were present in its parents. The simplest form of recombination is the 1-point crossover (De Jong (2006)). Suppose the candidate (a, b, c, d, e) was chosen as the first parent solution and (v, w, x, y, z) is the second parent solution. A crossover point i is chosen at random; the child solution is then created by taking genes, 1 to i from Parent 1 and genes $(i+1)$ to 5 from Parent 2. For example, for Parents 1 and 2 above, if the crossover point is chosen as $i = 3$ the resulting child solution would be (a, b, c, y, z). This can be extended to a general k-point crossover; for example, a 2-point crossover with crossover points 1 and 3 would result in the child solution (a, w, x, d, e).

De Jong (2006) notes that using a fixed number of crossover points can result in a bias; specifically genes that are located close together are more likely to be inherited together than those that appear further apart (De Jong (2006)). An alternative strategy is to select the number of crossover points at random. A common way to do this is to randomly decide which parent solution contributes its gene, independently, at each position in the child solution. If each parent is equally likely to contribute to every position, this is known as uniform crossover (De Jong (2006)). For example, if Parent 1 and Parent 2 are equally likely to contribute their gene to each entry in the child solution, we are essentially performing a coin flip at each position; if the outcome is "heads" Parent 1 will contribute their gene at that position, otherwise Parent 2 will make the contribution. For the outcome "HTHHT", the resulting child solution is (a, w, c, d, z). De Jong (2006) provides an informative discussion of the long-term behavior of various recombination strategies.

A good GA will allow for a balance between selection and exploration (i.e., reproduction). When a weaker selection strategy is used, strong and weak parents are chosen with similar

frequency, resulting in more heterogeneous offspring. In this instance, choosing mutation and recombination strategies that create offspring similar to their parents will help speed convergence. This could mean using a less variable mutation operator or performing recombination with fewer crossover points. When a stronger selection strategy is employed, the stronger individuals will reproduce more frequently resulting in more homogeneous group of offspring solutions. In this case, it is beneficial to choose mutation and recombination strategies that allow for more exploration of the search space by creating offspring that are quite different from their parents.

3.2 Implementation of a Genetic Algorithm

The goal of a resource allocation study is to find an optimal (or nearly-optimal) way to spend resources on a new experiment. To find such an allocation we implement a genetic algorithm; some of the details are presented here, with a more complete description in Appendix A. Suppose there are $k+1$ components that comprise the system (k basic components/subsystems and the full system). A candidate solution will be represented in the GA as an integer-valued vector of length $k+1$; the gene in the i^{th} position corresponds to the number of tests allocated to the i^{th} component. Let B represent the budget and $c_i, i = 1, \ldots, k+1$ represent the testing costs of each component.

3.2.1 Generating the Initial Population

The first step is to decide on the size of the population and how to generate random allocations that satisfy the budget constraints. Much of the literature suggests population sizes between 20 and 50 are effective (Michalewicz (1996), De Jong (2006)), thus we follow the lead of Hamada *et al* (2004) and use a population size of $m = 25$. To generate an allocation for the initial population we do the following:

1. Generate a vector $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_{k+1})$ from the $(k+1)$-dimensional Dirichlet distribution with parameter vector α=$(\frac{1}{k+1}, \frac{1}{k+1}, \ldots, \frac{1}{k+1})$. This simulated vector represents the percentage of the budget that will be spent on each component. To generate an observa-

tion from the $(k+1)$-dimensional Dirichlet distribution, we generate $k+1$ independent Gamma($\frac{1}{k+1}$,1) random variables (Y_i) and transform according to $Z_i = \frac{Y_i}{\sum Y_i}$.

2. Now, $D_i = B \times Z_i$ represents the actual (dollar) amount to be spent on Component i tests.

3. Find the number of tests to be performed on Component i as $n_{2,i} = \text{floor}(D_i/c_i)$.

This procedure is repeated until $m = 25$ allocations are generated for the initial population.

3.2.2 Parent Selection and Recombination

In this implementation, a rank-proportional selection criterion is employed to selection parent solutions for recombination. Even though this method of selection tends to increase selective pressure, which can lead to a homogeneous population too soon and increase the risk of convergence to a local optimum, it is preferred in this application as fitness-proportional selection is essentially equivalent to uniform selection. The essential equivalence is due to the fact that the calculated expected entropies typically do not display large differences in values (i.e., differences often appear in the second or third decimal place). Thus, the selection probability for Solution i with expected entropy f_i is $f_i / \sum_i f_i$ and there is nearly equal likelihood of parent selection for all population members. Using rank-proportional selection reduces the chance that weak individuals (poor solutions) produce children, resulting in overall stronger children (better solutions).

Once two parents have been selected from the current population for recombination, a uniform crossover is performed. For example, consider a system with a total of 5 components, in which Components 1-4 cost \$1 and Component 5 (the full system) costs \$2 and the overall budget is \$20. Suppose two parents are selected at random from the population according to their rank, with Parent 1 being the allocation (5,2,3,6,2) and Parent 2 being the allocation (0,0,0,0,10). If the i^{th} coin flip results in "heads" Parent 1 will contribute its i^{th} gene to the child, otherwise Parent 2 will make the contribution. For the sample coin flip "HHHTH", the resulting child solution would be (5,2,3,0,2). Even though this allocation does not use the entire

budget, it is still considered a feasible candidate. If the resulting candidate allocation exceeds the budget, it is repaired by selecting a component with a nonzero sample size at random and decreasing that component's sample size by one. This is repeated until the budget constraint is satisfied. The selection and recombination process is repeated until $m = 25$ children are created.

3.2.3 Mutation

Next, each parent in the population is mutated to create a set of $m = 25$ potential mutation solutions. For a single parent, mutation proceeds per the approach of Hamada *et al* (2004). Each entry of the candidate's allocation is mutated with probability $\exp(-\mu g)$, where μ is the mutation rate and g is the generation number; here, the mutation rate is specified as $\mu = 0.01$, though Michalewicz (1996) and Hamada *et al* (2004) note that changing this value will have little effect on the performance of the algorithm. The mutation occurs such that when the i^{th} entry is mutated, the expected value of the mutation is approximately equal to $n_{2,i}$ (the current sample size) for Component i and the variability in the mutations decreases with each generation. For Component i, Hamada *et al* (2004) define L_i as the minimum number of tests allowed on Component i and $U_i = floor(B/c_i)$, or the maximum number of tests allowed on Component i. Then for the current sample size $n_{2,i}$

$$z_i = \frac{n_{2,i} - L_i}{U_i - L_i}$$

and

$$d_i = \log[z_i/(1 - z_i)] + [Uniform(0, 1) - 0.5]\sigma\exp(-\mu g),$$

where σ controls the rate at which the variation decreases with generation (Hamada *et al* (2004)) and is specified here to be $\sigma = 1.25$. The mutated sample size for Component i at Generation g is

$$n'_{2,i} \quad = \quad \text{floor}\left(L_i + (U_i + 1 - L_i)\frac{\exp(d_i)}{1 + \exp(d_i)}\right). \tag{3.1}$$

This mutation is performed at each gene in the parent solution to form the mutation solution.

For illustration we once again consider the five component system described above, where tests on Components 1-4 cost \$1 each, tests on Component 5 cost \$2, and the budget is \$20. Consider the parent solution (5,2,3,6,2) and a mutation occurring during Generation 5. First apply the mutation operator to Component 1. The maximum number of tests possible on this component is $U_1 = 20$; the minimum is $L_1 = 0$. For this component, $z_1 = .25$. Suppose we draw $u = 0.005675764$ from $Uniform(0, 1)$, then the new sample size for Component 1 is $n_{2,1} = 3$. Suppose when applying the mutation operator to the remaining components we draw the random uniforms 0.37634045, 0.04160195, 0.8593304, and 0.51226843. The resulting mutated solution is (3,1,1,8,2). Now consider the same parent solution with a mutation occurring during Generation 100. Using the same random draws as above (for comparison's sake), the mutated solution is (4,1,1,7,2), which is slightly closer to the parent solution than was the mutated solution from Generation 5. As with offspring created via recombination, a candidate exceeding the budget constraint is repaired by iteratively selecting a component with a nonzero sample size at random and decreasing it's sample size by one until the constraint is satisfied.

At the end of a generation, there are $3m = 75$ potential solutions, those from the current population, the child solutions, and the mutation solutions. To determine the fitness of the children and mutation solutions (those in the current population have already been evaluated), the 50 solutions are evaluated as described in Chapter 2 using the results from the initial analysis performed via MCMC. Before proceeding to the next generation, this set of $3m = 75$ solutions is reduced to $m = 25$ using an elitist strategy in which the m most fit solutions (i.e., those with the lowest expected entropy) are chosen to survive. Using an elitist strategy ensures that solutions as good as, or better than, those in the current generation move on to the next generation (De Jong (2006)); that is, the best solution in Generation g will be no worse than the best solution in Generation $g - 1$. The algorithm is typically run for $G = 100$ generations. Additionally, the GA is run multiple times (typically three) for each problem. Multiple runs of the GA are utilized to help avoid selecting a local optima as the best allocation. Complete GA details are provided in Appendix A.

In this implementation we are initially assuming that candidate allocations are small enough

that they can be evaluated as described in Section 2.3; that is, the candidates are sufficiently small so that enumerating all outcomes from the candidate allocation to calculate the expected entropy is a feasible undertaking. Evaluating candidate allocations in this manner is deterministic; a candidate will have the same expected entropy each time it is evaluated. The amount of computation time necessary to run the GA is greatly reduced by recognizing this and not needlessly re-evaluating candidates. This can be done by checking to see if a candidate has already been evaluated; if it has been evaluated already, the previously recorded expected entropy is used for that candidate allocation every time it appears, otherwise the candidate is evaluated via complete enumeration of outcomes as in Section 2.3. In Section 3.4, modifications are made to accommodate large systems and allocations.

3.3 Example: Using Genetic Algorithms to Find an Optimal Resource Allocation

To illustrate how the methodology introduced in Chapter 2.3 can be combined with a genetic algorithm for finding an optimal resource allocation, consider the eight component series system display in Figure 3.1. Suppose first-stage data were collected at all levels of this series system, with 200 tests performed on all basic components (Components 1 - 5), 100 tests performed on the subsystems (Components 6 and 7), and 50 tests performed at the system level. The first-stage data, as well as the expert best guesses for the reliability of these components, are given in Table 3.1.

Table 3.1 First-stage data, expert best guesses, and component testing costs for eight component series system

Type	Component	Successes	Tests	π	Cost
Basic	1	200	200	.99	\$1
Basic	2	188	200	.95	\$1
Basic	3	199	200	.99	\$1
Basic	4	191	200	.95	\$1
Basic	5	196	200	.98	\$1
Subsystem	6	96	100	.9405	\$2
Subsystem	7	95	100	.9405	\$2
System	8	42	50	.8668	\$3

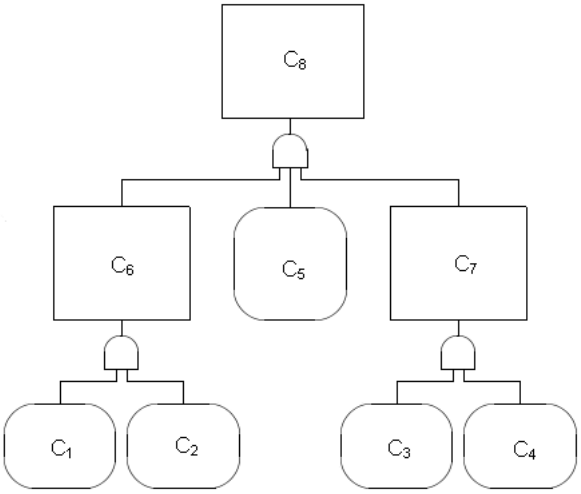


Figure 3.1 Event tree for eight component series system

The initial analysis was performed using the model of Johnson *et al* (2003) described in Section 1.2; 10,000 posterior draws were collected using a random-walk Metropolis-Hastings algorithm as described in Section 2.4. The system reliability posterior distribution is displayed in Figure 3.2. The system reliability posterior mean is 0.87213 and the posterior variance is about 0.000302. A central 95% credible interval for the system reliability posterior mean is (0.83629, 0.90425). For the bin width of $h \approx \frac{\hat{\sigma}}{10}$, 576 bins are used and the posterior entropy is calculated to be -2.64045.

Given the opportunity to collect second-stage data on the system, we would like to choose the allocation that would yield the largest expected gain in information, or equivalently the lowest expected entropy.

The component testing costs are displayed in Table 3.1 Tests on the basic components are the least expensive to perform and full system tests are the most expensive tests to perform. For a budget of $100, we wish to find the most informative allocation possible with these resources.

The genetic algorithm described in Section 3.2 was run for 50 generations. The optimal allocation found contains 47 tests of Component 2, 24 tests of Component 4, and 29 tests of Component 5, with no tests allocated for the remaining components; the expected entropy for

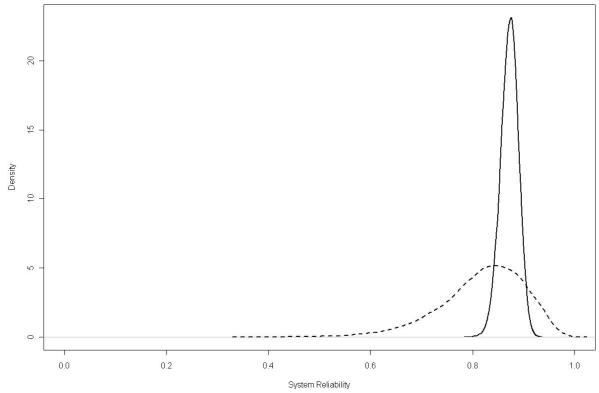


Figure 3.2 Posterior distribution of system reliability given first stage data

this allocation was computed to be -2.68523. This allocation is intuitive for several reasons. First, tests on the system and subsystems are more expensive than tests on the basic components, thus we are getting as many tests as possible for the available resources. Second, the chosen allocation uses the available resources for tests of the basic components that had the worst performance in the initial testing stage, with Component 2 having the apparently worst performance in the first-stage receiving the majority of the allocated tests in the second-stage. Figure 3.3 and Figure 3.4 detail the progress of the algorithm.

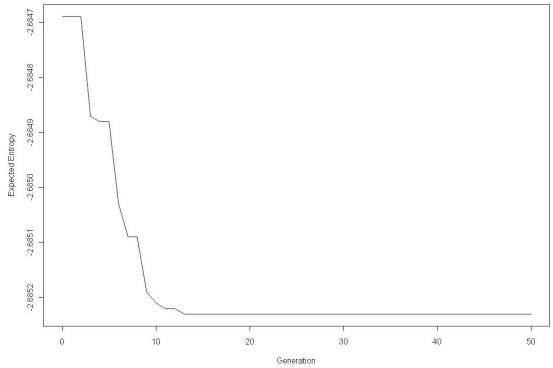


Figure 3.3 Best-so-far curve displays the expected entropy of the best allocation at each generation

The best-so-far curve in Figure 3.3 displays the expected entropy of the best allocation at each generation, including the initial population (Generation 0). Because the algorithm is elitist, the best-so-far curve is monotonic non-increasing; the best allocation from Generation

g will not be any worse than the best allocation in Generation $g-1$. The plot indicates that the best allocation was found at Generation 14. The composition of the best allocation, at each generation, is detailed in Figure 3.4.

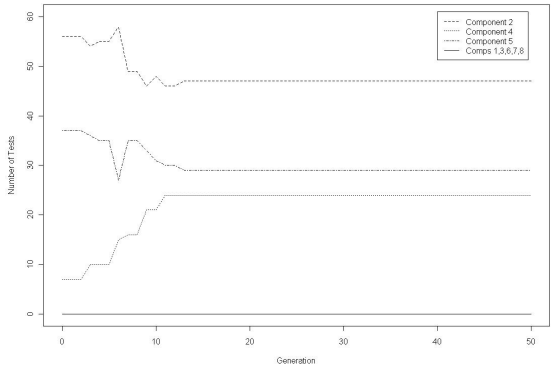


Figure 3.4 Tests per Component for the Best Allocation at Each Generation

Figure 3.4 tracks the composition of the best allocation at each generation in the GA. All lines (with the exception of the solid line at 0) represent a single component; the solid line represents Components 1, 3, 6, 7, and 8 as the best allocation always included 0 tests of the components. To verify that a very good, if not the best, allocation was chosen, the algorithm was run two more times. A second run of the GA resulted in a very similar allocation, (0, 49, 0, 22, 29, 0, 0, 0), with the same expected entropy as the first GA run, -2.68523. The best allocation found in the third GA run was (0, 51, 0, 20, 29, 0, 0, 0) with expected entropy -2.68521; this allocation is quite similar to those identified by the other runs of the GA.

The average computing time for these three GA runs was about 10.76 days, with a standard deviation of 5.03 days; the fastest run executed in 5.97 days while the longest run took roughly 16 days. The large variability in the computing time of these three runs is due to the size of the candidate allocations generated in the early generations of each run; that is, one run of the GA generated several quite large candidate allocations for the initial population (on the order of 1 million possible outcomes or more), while the other two runs generated fewer large allocations. When the full enumeration evaluation procedure (i.e., Section 2.3) is used, the computing time necessary to implement the GA depends heavily on the candidates that are generated by the

GA. In the following section, we described how to use the sampling-based candidate evaluation methodology from Section 2.6 to evaluate the candidate allocations generated by the GA; this use of the sampling-based candidate evaluation methodology allows large candidate allocations to be easily evaluated, and ultimately leads to shorter computing time for the GAs.

3.4 Genetic Algorithms for Large Systems and Allocations

With small systems and allocations (e.g., of the size of the example given in Section 3.3), we can evaluate the fitness of candidates by enumerating all outcomes for the candidate allocation and computing the expected entropy of the candidate. For large systems and allocations, it is infeasible to enumerate all possible outcomes in order to calculate the expected entropy. A sampling-based evaluation procedure performs quite well for evaluating candidate allocations, but introduces a potential complication when used to evaluate candidates in the genetic algorithm. Now that candidates are evaluated using an approximation, the question arises as to how precisely the candidates should be evaluated while searching for the optimal allocation. Because a stochastic evaluation procedure is being used to estimate the expected entropy, we re-evaluate all candidates each time they are generated by the GA, as repeated evaluations of the same allocation, while hopefully similar, will not be exactly equal.

Fitzpatrick and Grefenstette (1988) provide theoretical justification for increasing either the population size or number of generations, rather than the sample size, to improve the effectiveness of the genetic algorithm when candidates are evaluated using an approximation; in particular, they discuss the scenario in which candidates are evaluated using sampling. Fitzpatrick and Grefenstette (1988) define *hyperplanes* to be sets of solutions that have similar composition. For example, if solutions in the GA represent candidate allocations for collecting data from a five-component system, one hyperplane is the set of all solutions of the form (0, x, x, x, 0); this hyperplane is the set of all solutions that have no Component 1 or system tests. They argue that the quality of the GA's search depends on the variability associated with estimates of the performance (i.e., average expected entropy) of hyperplanes rather than with the performance of individual solutions (Fitzpatrick and Grefenstette (1988)); that is, reducing

the variability associated with the estimation of the performance of hyperplanes improves the GA's ability to distinguish between the areas of the search space (i.e., hyperplanes) that contain good solutions and those that do not.

Fitzpatrick and Grefenstette (1988) show that decreasing the sample size N (i.e., the number of possible outcomes sampled) does not result in much loss of accuracy in the estimation of the performance of hyperplanes. They note that for a fixed amount of computing time, the sample size can be decreased so that more candidates can be evaluated, either by increasing the population size or the number of generations. They conclude that in some cases, in particular when the cost of evaluating candidates is more expensive than the creation of new candidates, more effective searches can be conducted by decreasing the sample size and increasing either the population size or the number of generations. They measure the effectiveness of the search by considering the overall fitness of the final population in the GA.

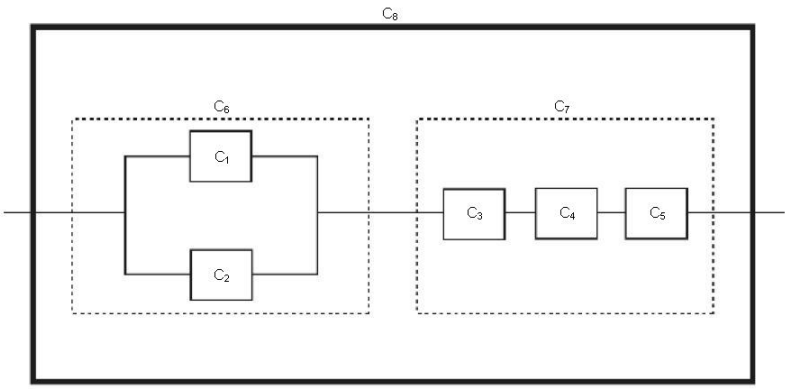


Figure 3.5 Simple eight component series/parallel system (Figure 9.2 from Hamada *et al* (2008))

3.4.1 Example: GA for Eight Component Series/Parallel System with a Single Expert

As an illustration, we consider the following example from Hamada *et al* (2008). They describe a simple eight component series/parallel system, the reliability block diagram for which is displayed in Figure 3.5. This system (labeled Component 8) contains two major subsystems

connected in series. The first subsystem (Component 6) is comprised of two components (1 and 2) connected in parallel while the second subsystem (Component 7) is made up of Components 3, 4 and 5 connected in series. Table 3.2 displays the first-stage data and expert guesses for the reliabilities of the components in the system, as well as the costs associated with testing the various components. Under the specified cost structure testing either subsystem has the same cost as testing all basic components that comprise that subsystem; similarly, testing the full system has the same cost as testing one of every basic component in the system.

Table 3.2 First-stage data, expert guesses, and testing costs for simple eight component series/parallel system (Hamada *et al* (2008))

Type	Component	Data	π	Cost
Basic	1	34/40	0.9	\$1
Basic	2	47/50	0.9	\$1
Basic	3	3/5	0.95	\$1
Basic	4	8/8	0.95	\$1
Basic	5	16/17	0.95	\$1
Subsystem	6		0.9	\$2
Subsystem	7	10/10	0.9	\$3
System	8	15/20	0.8	\$5

The model of Hamada *et al* (2008) is similar to that described in Section 1.2 in that they assume that binomial data and expert information are present, and represent non-basic component reliabilities in terms of basic component reliabilities. However, they use the expert information in a slightly different way to obtain prior distributions for component reliabilities.

For basic components, if an expert's best guess, π_i, is specified for Component i, the prior distribution for the reliability of this component is Beta($\tilde{n}_i\pi_i$, $\tilde{n}_i(1-\pi_i)$), where $\tilde{n}_i$ is, effectively, a sample size (Hamada *et al* (2008)). If a best guess is not specified for a basic component, a diffuse or non-informative prior is used (i.e., Jeffrey's prior or a Uniform prior).

If an expert best guess is specified for non-basic components (i.e., subsystems and system), the prior distribution for the reliability of that component has the form of a binomial likelihood

$$p_i^{\tilde{n}_i\pi_i}\,(1-p_i)^{\tilde{n}_i(1-\pi_i)}\,;$$

that is, the prior distribution for the reliability of non-basic components, if an expert best

guess π_i is specified, is Beta($\tilde{n}_i\pi_i + 1$, $\tilde{n}_i(1-\pi_i) + 1$). If an expert guess is not available for a non-basic component, that component does not have it's own prior distribution.

The precision parameters $\tilde{n}_i$ can be treated as either fixed constants or random quantities. An instance in which they might be treated as fixed is when the expert best guess π_i represents the success rate from prior testing and thus $\tilde{n}_i$ truly is a sample size (Graves and Hamada (2004)). If the $\tilde{n}_i$ are not treated as known and they are assumed to be the same for all components (i.e., $\tilde{n}_i = \tilde{n}$), a common prior distribution can be specified, such as $\tilde{n} \sim$ Gamma$(5, 1)$ (Hamada *et al* (2008)).

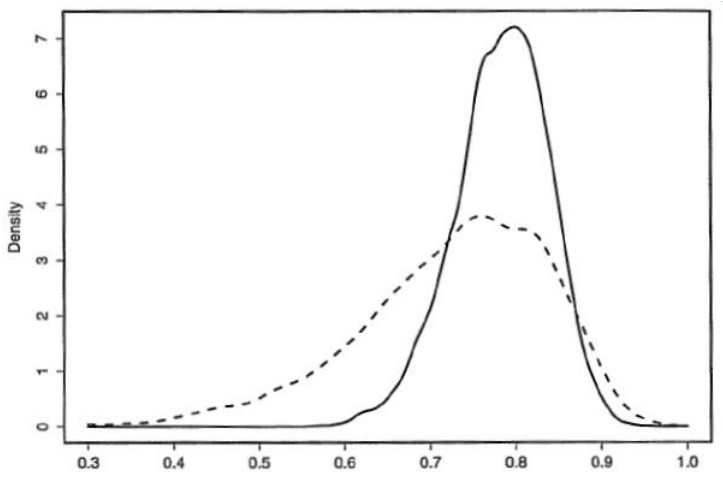


Figure 3.6 Prior (dashed) and posterior (solid) distributions for system reliability (Figure 9.3 from Hamada *et al* (2008))

The component reliability prior distributions are combined with the binomial likelihood using Bayes' Rule to obtain the posterior distribution of system reliability. The marginal system reliability posterior distribution obtained from their analysis is displayed in Figure 3.6 (this is Figure 9.3 from Hamada *et al* (2008)). They find the 90% credible interval for system reliability to be (0.689, 0.865); the length of the credible interval is 0.176.

After the initial analysis is complete, they try to find an optimal allocation of resources using the cost structure described in Table 3.2 and a budget of \$1000. They use the width of the 90% credible interval as their planning criterion. To find the 90% credible interval for a given candidate allocation, they first generate $N_d = 500$ datasets consistent with the initial

data (as described in Section 2.2) and then perform an analysis via MCMC for each of these artifical datasets to obtain the updated posterior distribution; $N_p = 2,000$ draws are collected from the updated posterior distribution. The width of the 90% credible interval is computed for each of the $N_d = 500$ updated posterior distributions. They then report an upper quantile of this empirical distribution of credible interval widths as the value of their planning criterion.

A genetic algorithm is used to find the optimal resource allocation. The GA implementation uses rank-proportional selection to choose parents for uniform crossover and the mutation strategy described in Section 3.2. They use a population size of $m = 20$ and run the GA for 50 generations. This GA implementation requires $20 + 40 \times 50 = 2,020$ candidate evaluations (20 evaluations for the initial population, and 20 crossover evaluations and 20 mutation evaluations in the subsequent generations), where each evaluation requires 500 analyses via MCMC.

The best allocation found by the GA, (0, 0, 208, 137, 128, 0, 175, 0), consists only of tests on Components 3, 4, 5, and 7 (i.e., the series subsystem and the basic components that comprise it). In the initial dataset, the fewest tests were performed on the components in this subsystem. The best allocation identified by the GA was found to have a 90% credible set width of 0.0725. Figure 3.7 tracks the number of tests per component for the life of the genetic algorithm.

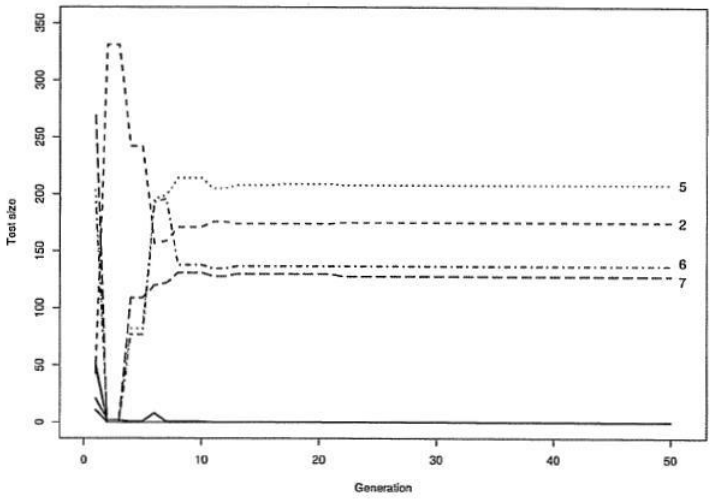


Figure 3.7 Tests per components at each generation (Figure 9.8 from Hamada *et al* (2008))

Hamada *et al* (2008) then re-evaluate the allocation obtained by the GA twice more using $N_d = 100,000$ generated datasets and $N_p = 50,000$ posterior draws. For these two runs, they find credible interval widths of 0.073358 and 0.073363 and thus take 0.0734 to be the "true" credible interval width for this allocation. Hamada *et al* (2008) note that a single test on Component 7 has the same cost as testing one of each of the basic components that comprise Component 7 (i.e., Components 3, 4, and 5). They then consider an allocation that proportionally gives the subsystem tests to its components, (0, 0, 439, 289, 270, 0, 0, 0). They evaluate this allocation twice using $N_d = 100,000$ generated datasets and $N_p = 50,000$ posterior draws. They find the 90% credible interval widths for these two runs to be 0.071439 and 0.071426, and thus find the "true" credible interval width for this allocation to be 0.0714, making this a better allocation than the one selected by the genetic algorithm. They note that the genetic algorithm did not find this allocation because the difference in the credible interval widths is within the simulation variability for the candidate evaluations.

Using the model of Johnson *et al* (2003) and collecting 10,000 draws from the posterior distribution (displayed in Figure 3.8), we estimate the system reliability posterior mean to be about 0.795 and the 90% credible interval to be (0.7069, 0.8726); the width of this credible interval, 0.166, is slightly narrower than that of Hamada *et al* (2008). The posterior variance is 0.00258, and thus the standard deviation is approximately 0.05. Using 200 bins, or $h \approx \frac{\hat{\sigma}}{10}$, the posterior entropy was computed to be -1.57966. The precision parameter N has posterior mean 6.942, indicating that, on average, the expert's opinion is worth roughly 7 system tests.

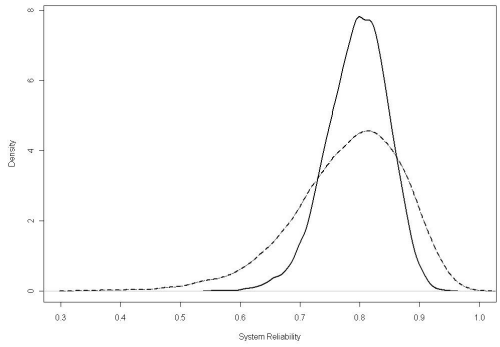
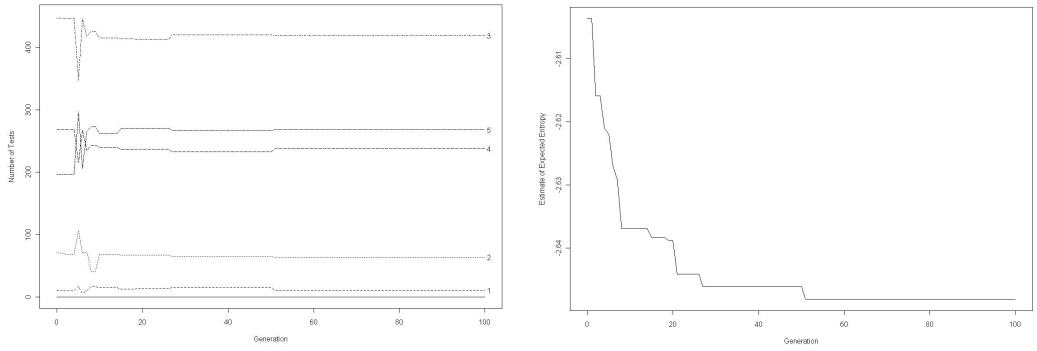


Figure 3.8 Prior (dashed) and posterior (solid) distributions of system reliability under model from Section 1.2

To search for an optimal allocation of resources in the problem described above, a GA with a population of size 40 was run, three times, for 100 generations. Recombination and mutation were carried out as described in Section 3.2. Each candidate allocation was evaluated by generating 1000 possible outcomes, calculating the entropy of the updated posterior distribution of system reliability given the initial data and each generated outcome, and then reporting the average of those calculated entropies. The three runs of the GA reached vastly different final populations; in two runs, the final populations had considerably lower average population fitness than the other run. In the best of these three runs, the selected allocation was (11, 64, 419, 238, 268, 0, 0, 0); the expected entropy was estimated to be -2.64814, with standard deviation 0.285712. The progress of the genetic algorithm is detailed in Figures 3.9(a) and 3.9(b). These plots indicate that this allocation was identified around Generation 50.



(a) Tests Per Component

(b) Estimated Expected Entropy for Best Allocation in Each Generation

Figure 3.9 Performance of a single run of a genetic algorithm to find an optimal resource allocation for the simple eight component system of Hamada *et al* (2008)

The 40 candidates in the final population of the genetic algorithm are displayed in Table 3.3, along with their estimated expected entropy and standard deviation based on samples of size 1000. Each of the 40 candidates used most of the resources to obtain more tests on Components 3 - 5 (i.e., those in the series subsystem), but allocated a small amount of resources to Components 1 and 2 (i.e., those in the parallel subsystem). Components 3 - 5 had the lowest first-stage sample sizes, and are quite inexpensive to test, so it is not surprising

that an informative candidate allocation would include many tests of those components. Both Components 1 and 2 saw several failures in the first-stage testing, and no first-stage tests were performed on the parallel subsystem (Component 6), so it is clear that some information will be gained by including these components in a second-stage experiment. These 40 candidates, as well as the two allocations described by Hamada *et al* (2008), were re-evaluated using samples of size 10,000; the estimated expected entropy, standard deviation, and 95% confidence intervals for the expected entropy for these allocations are also displayed in Table 3.3.

Even after being evaluated more precisely, there is still no clear separation between the expected entropy of the candidates in the final population of the genetic algorithm; that is, we cannot conclude that one candidate in the final population is better than another. However, all of the candidates in this final population appear to be more informative than the two allocations described by Hamada *et al* (2008), though this could be a result of either the model used or the use of entropy as the planning criterion. At any rate, it appears that an allocation using all resources on basic component tests, with the bulk of the tests performed on the basic components in Component 7 (the series subsystem) is a very informative candidate allocation.

The average computing time for these three GA runs was roughly 1.91 days, with a standard deviation of 0.53 days. Using the sampling-based candidate evaluation methodology, the computing times for each GA run shorter, and more consistent, as the size of the allocations generated by the GA no longer affects the computing time. That is, now the computing time depends on the number of outcomes sampled rather than the actual size of the candidate allocations.

Table 3.3 Estimated expected entropy and standard deviation for candidates in the final population of genetic algorithm as evaluated by the genetic algorithm, using samples of size 1000. These 40 allocations, as well as the two allocations described by Hamada *et al* (2008) were re-evaluated using samples of size 10,000

Allocation	$N = 1000$		$N = 10000$		
	$\hat{G}$	$s_{\hat{G}}$	$\hat{G}$	$s_{\hat{G}}$	95% CI for $\tilde{G}$
(11, 64, 419, 238, 268, 0, 0, 0)	-2.64814	0.285712	-2.62112	0.204295	(-2.62512, -2.61712)
(14, 64, 418, 237, 267, 0, 0, 0)	-2.64659	0.269201	-2.62519	0.217884	(-2.62946, -2.62092)
(15, 65, 420, 233, 267, 0, 0, 0)	-2.64609	0.270805	-2.62342	0.206277	(-2.62746, -2.61938)
(14, 65, 412, 237, 270, 0, 0, 0)	-2.64595	0.254789	-2.62394	0.216505	(-2.62818, -2.61970)
(11, 65, 419, 233, 267, 0, 0, 0)	-2.6459	0.265957	-2.62345	0.218565	(-2.62773, -2.61917)
(13, 65, 420, 236, 266, 0, 0, 0)	-2.64504	0.260433	-2.62147	0.203191	(-2.62545, -2.61749)
(11, 64, 420, 238, 267, 0, 0, 0)	-2.64492	0.253857	-2.62464	0.207843	(-2.62871, -2.62057)
(13, 67, 414, 236, 270, 0, 0, 0)	-2.64486	0.292032	-2.62258	0.204095	(-2.62658, -2.61858)
(15, 68, 415, 240, 262, 0, 0, 0)	-2.64476	0.278337	-2.62558	0.213509	(-2.62976, -2.62140)
(14, 67, 412, 237, 270, 0, 0, 0)	-2.64437	0.242631	-2.62571	0.209618	(-2.62982, -2.62160)
(15, 66, 419, 231, 269, 0, 0, 0)	-2.64431	0.246006	-2.62521	0.211131	(-2.62935, -2.62107)
(14, 67, 412, 237, 270, 0, 0, 0)	-2.64411	0.231632	-2.62375	0.201843	(-2.62771, -2.61979)
(11, 65, 420, 233, 267, 0, 0, 0)	-2.64367	0.251665	-2.62206	0.208272	(-2.62614, -2.61798)
(12, 66, 420, 233, 269, 0, 0, 0)	-2.64345	0.266606	-2.62277	0.212268	(-2.62693, -2.61861)
(13, 65, 411, 221, 290, 0, 0, 0)	-2.64331	0.265568	-2.62316	0.208924	(-2.62725, -2.61907)
(11, 65, 419, 237, 267, 0, 0, 0)	-2.6431	0.279387	-2.62139	0.206225	(-2.62543, -2.61735)
(15, 61, 420, 236, 268, 0, 0, 0)	-2.64288	0.286099	-2.62205	0.197947	(-2.62593, -2.61817)
(15, 65, 420, 233, 267, 0, 0, 0)	-2.64288	0.263409	-2.62313	0.202809	(-2.62711, -2.61915)
(15, 65, 420, 233, 267, 0, 0, 0)	-2.64276	0.250465	-2.62606	0.215053	(-2.63028, -2.62184)
(13, 65, 420, 236, 266, 0, 0, 0)	-2.64276	0.23716	-2.62304	0.205118	(-2.62706, -2.61902)
(13, 62, 420, 238, 267, 0, 0, 0)	-2.64226	0.238701	-2.62302	0.207845	(-2.62709, -2.61895)
(15, 64, 419, 234, 268, 0, 0, 0)	-2.64223	0.265084	-2.6238	0.215221	(-2.62802, -2.61958)
(15, 65, 419, 239, 262, 0, 0, 0)	-2.6422	0.252341	-2.62398	0.203418	(-2.62797, -2.61999)
(12, 66, 420, 233, 269, 0, 0, 0)	-2.64214	0.26033	-2.62499	0.214691	(-2.62920, -2.62078)
(13, 65, 418, 238, 266, 0, 0, 0)	-2.64182	0.247264	-2.62417	0.208947	(-2.62827, -2.62007)
(11, 65, 419, 238, 267, 0, 0, 0)	-2.64177	0.230913	-2.62077	0.201776	(-2.62472, -2.61682)
(15, 63, 417, 235, 270, 0, 0, 0)	-2.64176	0.258168	-2.62224	0.200798	(-2.62618, -2.61830)
(15, 64, 419, 233, 268, 0, 0, 0)	-2.6417	0.24795	-2.62798	0.222647	(-2.63234, -2.62362)
(14, 65, 412, 237, 270, 0, 0, 0)	-2.64169	0.243306	-2.6204	0.199252	(-2.62431, -2.61649)
(15, 65, 420, 233, 267, 0, 0, 0)	-2.64166	0.25624	-2.627	0.21738	(-2.63126, -2.62274)
(13, 67, 414, 236, 270, 0, 0, 0)	-2.64138	0.247044	-2.62254	0.204473	(-2.62655, -2.61853)
(15, 65, 420, 233, 267, 0, 0, 0)	-2.64128	0.246938	-2.62228	0.203727	(-2.62627, -2.61829)
(11, 64, 419, 238, 268, 0, 0, 0)	-2.64124	0.225288	-2.62469	0.203946	(-2.62869, -2.62069)
(15, 64, 419, 233, 268, 0, 0, 0)	-2.64121	0.263949	-2.61997	0.199191	(-2.62387, -2.61607)
(15, 65, 415, 240, 262, 0, 0, 0)	-2.64119	0.240097	-2.62113	0.209746	(-2.62524, -2.61702)
(14, 65, 419, 232, 270, 0, 0, 0)	-2.64111	0.254944	-2.62534	0.211414	(-2.62948, -2.62120)
(14, 65, 419, 232, 270, 0, 0, 0)	-2.64108	0.23222	-2.62267	0.196425	(-2.62652, -2.61882)
(15, 63, 417, 235, 270, 0, 0, 0)	-2.64101	0.236472	-2.6266	0.214104	(-2.63080, -2.62240)
(15, 64, 418, 239, 264, 0, 0, 0)	-2.64074	0.229616	-2.62241	0.205023	(-2.62643, -2.61839)
(11, 66, 419, 240, 264, 0, 0, 0)	-2.64071	0.253757	-2.62359	0.21291	(-2.62776, -2.61942)
(0, 0, 208, 137, 128, 0, 175, 0)	—	—	-2.54677	0.131175	(-2.54934, -2.54420)
(0, 0, 439, 289, 270, 0, 0, 0)	—	—	-2.59819	0.158853	(-2.60130, -2.59508)

3.4.2 Example: GA for Eight Component Series/Parallel System with Two Experts

We now consider a modification of the previous example in which expert best guesses are available from two different experts. The system structure, data, and cost structure are the same as those described in Figure 3.5 and Table 3.2. The best guesses for each expert are displayed in Table 3.4. In this scenario, the experts agree about the reliability of most components, but disagree about the reliability of both Components 1 and 2.

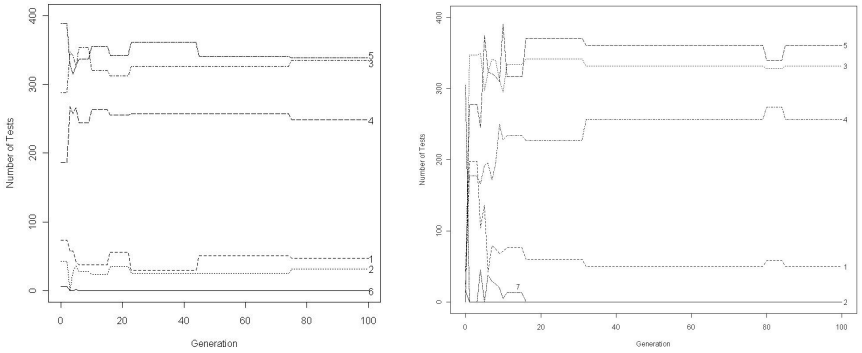
Table 3.4 Expert best guesses from two experts

Component	$\pi_{i,1}$	$\pi_{i,2}$
1	0.9	0.7
2	0.9	0.7
3	0.95	0.95
4	0.95	0.95
5	0.95	0.95
6	0.9	0.9
7	0.9	0.9
8	0.8	0.8

Under the model of Johnson *et al* (2003), which accommodates best guesses from multiple experts, the system reliability posterior mean was estimated to be 0.769 with posterior standard deviation roughly 0.047; the 90% credible interval for system reliability is (0.6884, 0.8431), which has width 0.1547. Using 200 bins, the posterior entropy was calculated to be -1.64593. Perhaps because of the discrepancy between experts, the posterior mean is slightly lower than that when guesses were available from only one expert. The precision parameter for the first expert, N_1, was estimated to have posterior mean 6.08 and that for the second expert, N_2, was estimated to be 5.88. Under this scenario, the experts' guesses are worth roughly 6 system tests.

A genetic algorithm was used to search for an optimal resource allocation for the cost structure in Table 3.2 and a budget of \$1000. The GA had a population size of 50 solutions and run for 100 generations. The candidate solutions were evaluated by drawing 1000 outcomes for each allocation and computing an estimate of the expected entropy as described in Section 2.6.

This GA was implemented three times; one run resulted in a final population of candidate solutions that were clearly not as informative as those in the final populations of the other two runs and is thus omitted here. The composition of the best allocation in one run of the GA is detailed in Figure 3.10(a).



(a) Tests Per Component for first run of GA

(b) Tests Per Component for second run of GA

Figure 3.10 Composition of the best allocations found in two runs of a genetic algorithm to find an optimal resource allocation for the simple eight component system when best guesses were specified by two experts

The first run of the GA found the solution (47, 32, 335, 248, 338, 0, 0, 0); the expected entropy of this allocation was estimated to be -2.55958, with standard deviation 0.190505. This solution allocates more tests to Component 1 and considerably fewer tests to Component 3 than that found when there was only information from a single expert.

The composition of the best allocation in the second run of the GA is detailed in Figure 3.10(b). One run of the GA identified the best allocation around Generation 30 and the second run around Generation 70, but the best allocations in the early generations of the second run appeared to be drastically different from the allocation that was ultimately selected. The second run found the solution (50, 0, 332, 257, 361, 0, 0, 0) with the expected entropy estimated to be -2.55732 and standard deviation 0.179761. Again, more tests are allocated to Component 1 while fewer tests are allocated to Component 3 than in the example in Section 3.4.1.

The unique candidates in the two final populations were re-evaluated by drawing samples

of size 10,000, and the estimated expected entropy, standard deviation, and 95% confidence interval are displayed in Table 3.5. When these 63 candidates were re-evaluated more precisely, (56, 35, 312, 257, 340, 0, 0, 0) was found to have the lowest expected entropy estimate; the allocation identified by the GA was ranked 4^{th} when the candidates were re-evaluated. Further, the confidence interval, constructed to account for the variability in the sampling-based evaluation, does not overlap with those for allocations 41 - 63, indicating that this allocation is significantly better than allocations 41 - 63. We can conclude that this allocation, or any of those in the top 40, should be a good choice for a second-stage experiment. Additionally, the allocation found by the GA in the example in Section 3.4.1, (11, 64, 419, 238, 268, 0, 0, 0), was re-evaluated under this scenario using a sample of size 10,000; the expected entropy of this allocation was estimated to be -2.54272, with standard deviation 0.178044. The 95% confidence interval for the expected entropy of this allocation was computed to be (-2.54621, -2.53923), which is significantly less informative than the allocations in Table 3.5. The allocations chosen by the GA under this scenario allocate more resources to Component 1, and fewer to Component 3, than those identified by the GA when there was only guesses specified by a single expert; the difference in the composition of these allocations may be due to dissenting expert opinions.

These three GA runs had an average compute time of 2.62 days, with standard deviation 0.59 days. This is slightly longer than that for the example in Section 3.4.1 (because a larger population size was used), though both examples in which the sampling-based candidate evaluation procedure was used had fairly short average computation times, indicating that this approach is quite computationally efficient. In Chapter 4 we use genetic algorithms for find optimal resource allocations for two larger systems.

Table 3.5 The 63 unique candidate allocations in the final populations of two runs of the GA were re-evaluated using samples of size 10,000 and the estimate of the expected entropy, the standard deviation, and a 95% confidence interval for the expected entropy of the allocation were computed

	$(C_1, C_2, C_3, C_4, C_5)$	$\hat{G}$	$s_{\hat{G}}$	95% CI		$(C_1, C_2, C_3, C_4, C_5)$	$\hat{G}$	$s_{\hat{G}}$	95% CI
1	(56, 35, 312, 257, 340)	-2.56467	0.205697	(-2.568702, -2.560638)	33	(55, 29, 335, 263, 315)	-2.55896	0.189733	(-2.562679, -2.555241)
2	(56, 35, 312, 263, 331)	-2.56463	0.202781	(-2.568605, -2.560655)	34	(35, 26, 345, 255, 339)	-2.55859	0.179031	(-2.562099, -2.555081)
3	(48, 23, 336, 254, 339)	-2.56456	0.196468	(-2.568411, -2.560709)	35	(38, 0, 329, 275, 358)	-2.558	0.176112	(-2.561452, -2.554548)
4	(47, 32, 335, 248, 338)	-2.56394	0.202035	(-2.567900, -2.559980)	36	(51, 0, 330, 275, 344)	-2.55744	0.174484	(-2.560860, -2.554020)
5	(51, 26, 323, 254, 340)	-2.56335	0.200514	(-2.567280, -2.559420)	37	(37, 0, 336, 256, 371)	-2.55743	0.175435	(-2.560869, -2.553991)
6	(37, 32, 328, 252, 351)	-2.56304	0.191926	(-2.566802, -2.559278)	38	(53, 0, 336, 263, 348)	-2.55732	0.183344	(-2.560914, -2.553726)
7	(34, 28, 315, 255, 368)	-2.5627	0.1923	(-2.566469, -2.558931)	39	(58, 0, 326, 274, 342)	-2.55727	0.197907	(-2.561149, -2.553391)
8	(55, 28, 331, 244, 342)	-2.56259	0.20353	(-2.566579, -2.558601)	40	(41, 0, 331, 278, 350)	-2.55712	0.18091	(-2.560666, -2.553574)
9	(56, 35, 312, 255, 340)	-2.56253	0.204706	(-2.566542, -2.558518)	41	(47, 0, 326, 285, 342)	-2.55708	0.179305	(-2.560594, -2.553566)
10	(35, 24, 347, 263, 331)	-2.5624	0.188906	(-2.566103, -2.558697)	42	(55, 0, 329, 254, 362)	-2.55684	0.192701	(-2.560617, -2.553063)
11	(56, 35, 312, 255, 342)	-2.56227	0.210409	(-2.566394, -2.558146)	43	(36, 0, 336, 278, 350)	-2.55663	0.172289	(-2.560007, -2.553253)
12	(30, 26, 326, 257, 361)	-2.56215	0.179885	(-2.565676, -2.558624)	44	(53, 0, 334, 278, 335)	-2.5565	0.178211	(-2.559993, -2.553007)
13	(51, 30, 312, 246, 361)	-2.56206	0.195155	(-2.565885, -2.558235)	45	(50, 0, 332, 257, 361)	-2.55636	0.176823	(-2.559826, -2.552894)
14	(46, 30, 334, 253, 337)	-2.56202	0.196882	(-2.565879, -2.558161)	46	(48, 0, 329, 268, 355)	-2.55577	0.171046	(-2.559123, -2.552417)
15	(44, 34, 330, 254, 337)	-2.56177	0.191996	(-2.565533, -2.558007)	47	(43, 0, 326, 273, 358)	-2.55543	0.168982	(-2.558742, -2.552118)
16	(51, 16, 338, 255, 340)	-2.56151	0.187687	(-2.565189, -2.557831)	48	(52, 0, 334, 255, 359)	-2.55536	0.173062	(-2.558752, -2.551968)
17	(46, 23, 341, 254, 336)	-2.56144	0.183502	(-2.565037, -2.557843)	49	(58, 0, 328, 274, 340)	-2.55536	0.183055	(-2.558948, -2.551772)
18	(55, 26, 326, 253, 340)	-2.56137	0.192255	(-2.565138, -2.557602)	50	(51, 0, 333, 273, 343)	-2.55454	0.173691	(-2.557944, -2.551136)
19	(34, 32, 346, 248, 338)	-2.56126	0.19208	(-2.565025, -2.557495)	51	(48, 0, 326, 270, 356)	-2.55447	0.167685	(-2.557757, -2.551183)
20	(56, 35, 312, 255, 342)	-2.56111	0.198769	(-2.565006, -2.557214)	52	(49, 0, 330, 275, 346)	-2.55418	0.175206	(-2.557614, -2.550746)
21	(49, 33, 323, 254, 341)	-2.56109	0.186255	(-2.564741, -2.557439)	53	(51, 0, 336, 255, 358)	-2.55414	0.170827	(-2.557488, -2.550792)
22	(33, 29, 335, 263, 338)	-2.561	0.187199	(-2.564669, -2.557331)	54	(49, 0, 336, 256, 359)	-2.55411	0.178688	(-2.557612, -2.550608)
23	(47, 34, 330, 248, 338)	-2.56098	0.198701	(-2.564875, -2.557085)	55	(46, 0, 327, 287, 340)	-2.554	0.178562	(-2.557500, -2.550500)
24	(37, 26, 328, 257, 351)	-2.56095	0.185758	(-2.564591, -2.557309)	56	(57, 0, 329, 255, 359)	-2.55389	0.174856	(-2.557317, -2.550463)
25	(47, 26, 326, 257, 340)	-2.56048	0.195197	(-2.564306, -2.556654)	57	(59, 0, 338, 258, 345)	-2.5535	0.170663	(-2.556845, -2.550155)
26	(49, 33, 312, 255, 342)	-2.56048	0.196516	(-2.564332, -2.556628)	58	(57, 0, 326, 278, 339)	-2.55302	0.173564	(-2.556422, -2.549618)
27	(28, 26, 324, 263, 359)	-2.56027	0.176689	(-2.563733, -2.556807)	59	(54, 0, 330, 274, 342)	-2.55294	0.165709	(-2.556188, -2.549692)
28	(34, 29, 346, 262, 329)	-2.56019	0.18349	(-2.563786, -2.556594)	60	(46, 0, 361, 257, 336)	-2.55267	0.172029	(-2.556042, -2.549298)
29	(51, 26, 326, 257, 340)	-2.55991	0.185862	(-2.563553, -2.556267)	61	(54, 0, 339, 258, 345)	-2.55228	0.169008	(-2.555593, -2.548967)
30	(33, 29, 346, 263, 329)	-2.55986	0.18392	(-2.563465, -2.556255)	62	(54, 0, 339, 238, 369)	-2.55225	0.170049	(-2.555583, -2.548917)
31	(52, 31, 334, 249, 334)	-2.5596	0.187046	(-2.563266, -2.555934)	63	(60, 0, 342, 227, 371)	-2.55187	0.171777	(-2.555237, -2.548503)
32	(47, 31, 335, 248, 339)	-2.55939	0.182778	(-2.562972, -2.555808)					

CHAPTER 4. CASE STUDIES

In Chapter 2 we introduced computationally efficient methodology that can be used to evaluate candidate allocations in resource allocation studies. In Chapter 3 we described how this methodology could be combined with genetic algorithms to find an optimal, or nearly optimal, allocation of resources. In this chapter we describe two system reliability studies and illustrate how the methods from the previous chapters can be efficiently used to find optimal resource allocations for real systems. The first case study addresses resource allocation for the series missile system described in Martz *et al* (1988). A notable feature of this system is that the subsystems that comprise the system are not all physically part of the missile, and tests on these subsystems, and the system itself, are not possible. The second case study involves resource allocation for a safety feature in a certain type of nuclear reactor. This application poses a challenge in that the system is extremely reliable and the system reliability posterior distribution exhibits very little variability.

4.1 Air-to-Air Heat-Seeking Missile

Martz *et al* (1988) describe a study in which the goal was to estimate the reliability of a certain air-to-air heat seeking missile under specific use conditions. The missile system is a series system consisting of five major subsystems; the event tree for this missile system is displayed in Figure 4.1. There are a total of 38 components in the system, 32 of which are basic components. Table B.1 in Appendix B describes the five subsystems and the basic components comprising them. Martz *et al* (1988) generically label the components that comprise the warhead (Component 33) as A-I for security classification reasons.

The available information included current binomial test data on some components in the

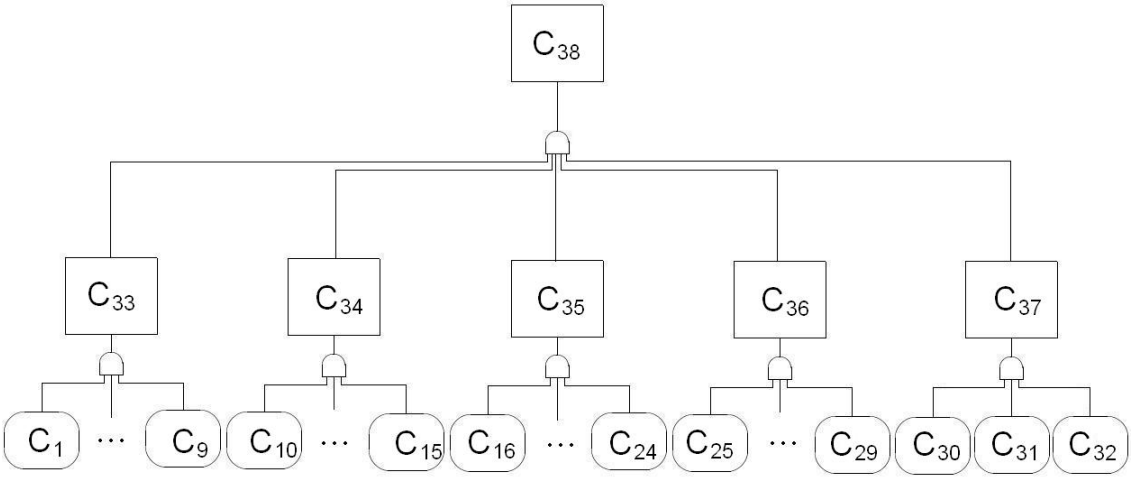


Figure 4.1 Event tree for series missile system

missile, as well as prior data at various system levels; we shall refer to the prior data of Martz *et al* (1988) as "historical" data. Some components, including the full system, had no current binomial data available. One such component is the subsystem labeled "Maintenance/Logistics" (Component 37); this subsystem is not a physical component of the missile, but rather represents "outside" factors such as handling and storage. The warhead subsystem and its components had no available historical data. The current binomial test data and historical data for the components in this system are provided in Table 4.1; Martz *et al* (1988) note that some of the data were deliberately altered to avoid security classification problems.

Martz *et al* (1988) use a two-stage analysis to find the posterior distribution of the missile system's reliability. Their analysis is outlined here, with more details provided in Appendix B. The first stage of the analysis is performed at the subsystem level as follows.

1. For each basic component in the system, the historical data for that basic component are used to form a prior distribution for the reliability of that basic component. The current binomial test data for that basic component are then used to update the basic component's reliability prior distribution to obtain the corresponding basic component reliability posterior distribution.

2. Suppose Subsystem i is comprised of k_i basic components connected in series. The

Table 4.1 First-stage data and expert best guesses for series missile system (Martz *et al* (1988))

C	Type	Data	π	C	Type	Data	π
1	Basic	30/30		20	Basic	130/130	357/360
2	Basic	80/80		21	Basic	247/250	254/257
3	Basic	39/40		22	Basic	129/130	250/252
4	Basic	30/30		23	Basic	249/250	250/252
5	Basic	90/90	846/848	24	Basic	330/330	341/352
6	Basic	10/10		25	Basic		797/802
7	Basic	29/30		26	Basic		796/802
8	Basic	20/20		27	Basic		792/802
9	Basic	5/5		28	Basic		791/802
10	Basic	50/50	399/402	29	Basic		386/402
11	Basic	50/50	278/302	30	Basic		1026/1122
12	Basic	99/100	1098/1102	31	Basic		1087/1092
13	Basic	23/25	654/690	32	Basic		1084/1092
14	Basic	50/50	299/302	33	Subsystem	8/8	
15	Basic	55/55	348/352	34	Subsystem	7/8	
16	Basic	129/130	246/250	35	Subsystem	191/205	258/271
17	Basic	130/130	245/250	36	Subsystem		56/68
18	Basic	129/130	247/250	37	Subsystem		
19	Basic	129/130	271/276	38	System		116/267

Subsystem i reliability *induced* prior distribution is the product of the k_i basic component reliability posterior distributions.

3. The Subsystem i reliability *native* prior distribution is formed from the Subsystem i historical data. If there are no historical data available for Subsystem i, there is no native prior distribution for Subsystem i.

4. Together the Subsystem i reliability induced and native prior distributions yield the Subsystem i *combined* prior distribution (i.e., the Subsystem i reliability combined prior distribution is based on the historical data for Subsystem i and its basic components, as well as the current binomial test data for the basic components in Subsystem i).

5. The current binomial test data for Subsystem i are used to update the Subsystem i reliability combined prior distribution to yield the Subsystem i reliability posterior distribution. (Steps 2 - 5 are repeated for all subsystems.)

The second stage of the analysis occurs at the system level and is performed similarly.

The subsystem and system combined prior distributions used by Martz *et al* (1988) are

displayed in Figure 4.2 (dashed lines). The combined prior distributions used by Martz *et al* (1988) are relatively informative, and the subsystem combined prior distributions are not influenced by the system level historical data, which suggests that the overall system reliability is around 0.43. The least informative subsystem reliability combined prior distribution used by Martz *et al* (1988) is that for the reliability of the warhead subsystem (Component 33). Historical data for most of the basic components in this subsystem were not available; the Jeffrey's prior distribution was used as the prior distribution for the reliability of each basic component for which historical data were not available.

The posterior distributions for the reliability of the subsystems and system are shown in Figure 4.3 (dashed lines). As there were no binomial data available for the C^3I and Maintenance/Logistics subsystems and the full system (Components 36, 37, and 38, respectively), the combined prior distributions and posterior distributions are the same for the reliabilities of these components. Martz *et al* (1988) find the system reliability posterior mean to be approximately 0.46. The standard deviation of the posterior distribution is 0.037 and the posterior entropy is -1.871771.

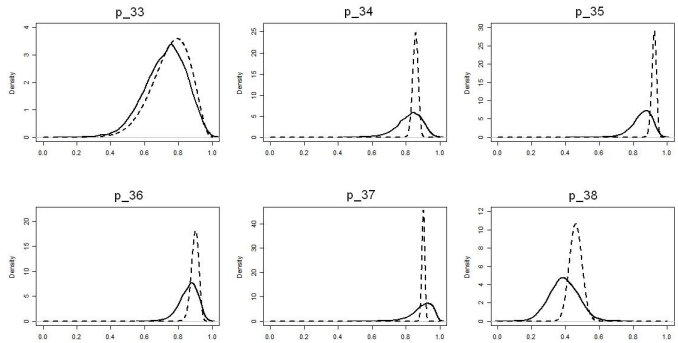


Figure 4.2 Prior distributions for the reliability of the five subsystems and missile system (solid = Johnson model, dashed = combined prior distributions of Martz *et al* (1988))

We re-analyze these data using the method of Johnson *et al* (2003) described in Section 1.2; the main results of the analysis are presented here while the details of the analysis are provided in Appendix B. The prior distributions for the five subsystems and full system are displayed

in Figure 4.2 (solid lines). In most cases, the combined prior distributions used by Martz *et al* (1988) are more informative than those specified in the method of Johnson *et al* (2003). A major difference between the method of Martz *et al* (1988) and that of Johnson *et al* (2003) is the manner in which the prior distributions are specified. The combined prior distribution for the reliability of any component under the method of Martz *et al* (1988) only uses historical data for that component, or for lower-level components in the event tree; for example, the historical data available at the system level do not influence the prior distributions used for the reliability of any basic components or the combined prior distributions for the subsystem reliabilities. In contrast, the method of Johnson *et al* (2003) borrows historical data from all levels of the system to obtain prior distributions for the reliability of all basic components, subsystems and the full system.

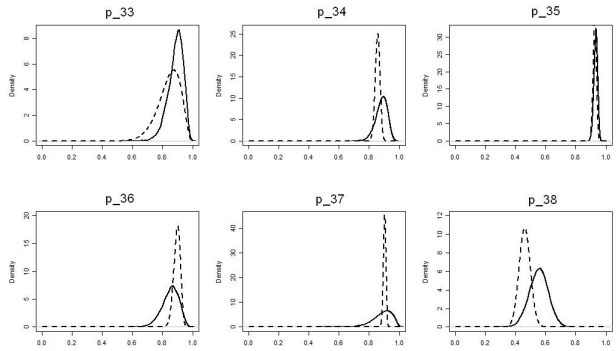


Figure 4.3 Posterior distributions for the reliability of the five subsystems and system (solid = Johnson model, dashed = Martz *et al* (1988))

The posterior distributions for the five subsystems and system reliabilities, based on the Johnson *et al* (2003) model, are provided in Figure 4.3 (solid lines). Even though there are no current binomial data at the system level, the available basic component and subsystem binomial data are used to update the system reliability prior distribution to obtain the system reliability posterior distribution. These data imply that the missile system is more reliable than the historical data had suggested. We find the system reliability posterior mean to be roughly 0.55. The standard deviation of the posterior distribution is 0.063, nearly twice the

standard deviation found by Martz *et al* (1988). Using bin width $h \approx \frac{\hat{\sigma}}{10}$, or 159 bins, the posterior entropy was computed to be -1.34971. The 95% credible interval for system reliability is (0.4276, 0.674).

Table 4.2 Hypothetical testing costs for testable components in missile system

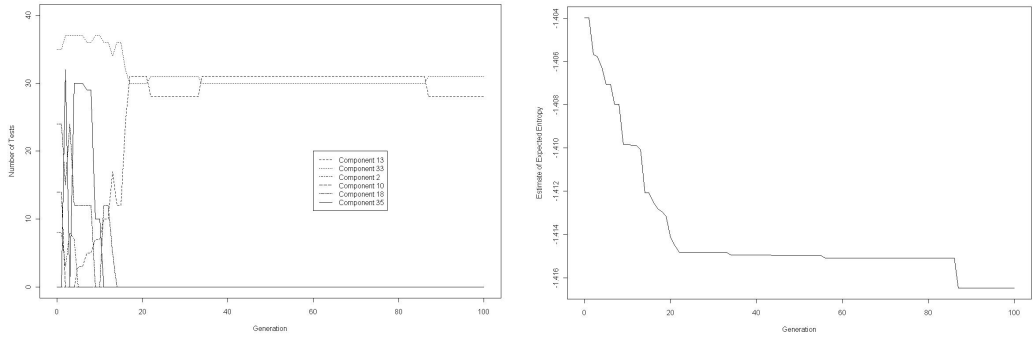
Component	Cost	Component	Cost	Component	Cost
Warhead (33)	\$125	Missile (34)	\$125	Aircraft (35)	\$5
A (1)	\$33	Power Supply (10)	\$20	Flight Structure (16)	\$7
B (2)	\$12	Guidance Sys. (11)	\$20	Avionics (17)	\$7
C (3)	\$25	Motor (12)	\$10	Power (18)	\$7
D (4)	\$33	Flight Structure (13)	\$40	Flight Control (19)	\$7
E (5)	\$11	Aircraft Interface (14)	\$20	Environmental (20)	\$7
F (6)	\$100	Control (15)	\$18	Acquisition (21)	\$4
G (7)	\$33			Launching (22)	\$7
H (8)	\$50			Missile Interface (23)	\$4
I (9)	\$200			Human Intervention (24)	\$3

Next we consider the problem of finding an optimal allocation for a specified second-stage budget. We assume that the only components that are testable are those for which there were binomial first-stage data. A hypothetical cost scenario is described in Table 4.2; these costs were obtained by assuming that approximately \$1000 was spent per component on initial tests and thus the cost of testing a component is inversely proportional to the number of tests performed. We assume that a budget of \$5000 is available to collect second-stage data, which is roughly $\frac{1}{6}$ of the first-stage budget implied by this cost structure. A genetic algorithm with a population size of $m = 30$ was used to search for an optimal allocation of resources. The GA evaluated candidate allocations by generating a random sample of 1000 outcomes for the candidate and estimating the expected entropy, using bin width $h \approx \frac{\hat{\sigma}}{10}$, as described in Section 2.6.

The genetic algorithm was run three times; two runs ended with similar final populations, while the third focused on an area of the solution space that was considerably less informative than the other two. Figures 4.4(a) and 4.4(b) display the performance of the first run of the GA. In this run, best allocation found by the GA allocated all resources to Components 13

and 33. Component 13 is the flight structure in the missile subsystem; Component 13 had the lowest first-stage success rate of all basic components, with 23 successes out of 25 tests. Component 33 is the warhead; the warhead subsystem and its components had the weakest prior information, as no expert's best guess was specified about most of these components. Figure 4.4(a) indicates that, in early generations, the GA considered allocations including tests of other components, but ultimately utilizing the entire budget for tests of Components 13 and 33 was found to be most informative.

An interesting artifact of how this cost structure was derived is that tests of one component in the warhead subsystem, specifically Component 9, cost more than tests of the warhead subsystem. As a result, the least expensive, and probably most informative, information about the reliability of the warhead subsystem comes from testing the subsystem itself. The allocation found by the GA consisted of 28 tests of the missile flight structure, at \$40 each, and 31 tests of the warhead subsystem, at \$200 each; the total cost of this allocation is \$4995. The estimate of the expected entropy for this allocation was -1.41648; the standard deviation, based on a sample of size 1000, was computed to be 0.0573.



(a) Number of tests per component

(b) Estimated expected entropy for the best candidate allocation found in each generation

Figure 4.4 Performance of a single run of the genetic algorithm for finding an optimal allocation of resources in the missile case study under the first cost structure

The candidates in the final population of the first run of the GA, displayed in Table 4.3, were quite similar. All candidates in the final population used the bulk of the resources on Components 13 and 33; some candidates also allocated a single test to one or more other

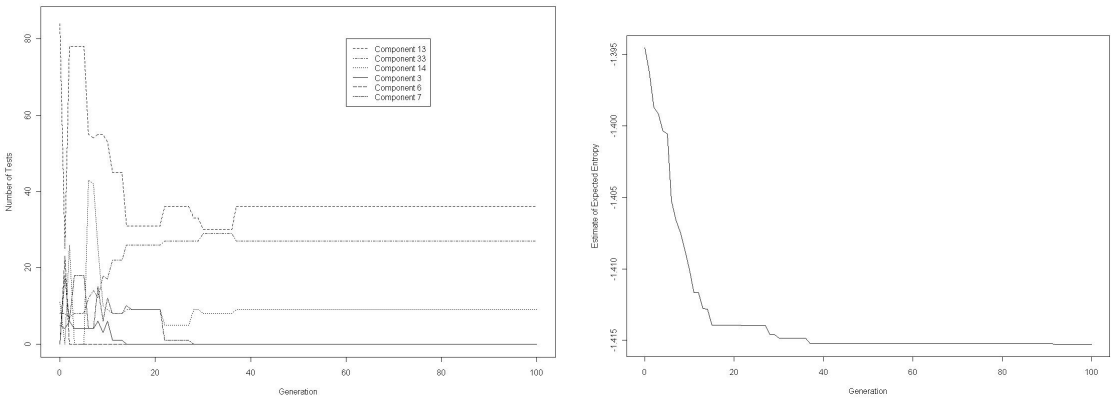
components. The estimated expected entropy, standard deviation, and a 95% normal-theory confidence interval for the expected entropy of each candidate allocation in the final population are also provided in Table 4.3. Using the sampling-based candidate evaluation procedure, the confidence intervals for each pair of candidates in the final population overlap; thus, based on these evaluations, no single candidate in the final population is clearly better than the rest.

Table 4.3 Candidate allocations in the final population of the first run of the GA

Flight Structure	Warhead	Aircraft Components				$\hat{G}$	$s_{\hat{G}}$	95% CI
13	33	21	22	23	24			
28	31	0	0	0	0	-1.41648	0.0572863	(-1.420031, -1.4129294)
31	30	0	1	0	1	-1.4156	0.0438963	(-1.418321, -1.412879)
31	30	0	1	0	0	-1.4151	0.04878	(-1.418123, -1.412077)
31	30	0	0	0	1	-1.41509	0.0647339	(-1.419102, -1.411078)
31	30	0	1	0	0	-1.41498	0.0304735	(-1.416869, -1.413091)
31	30	0	1	0	1	-1.41496	0.0438993	(-1.417681, -1.412239)
31	30	0	0	0	1	-1.41495	0.0529392	(-1.418231, -1.411669)
34	29	0	0	0	0	-1.41492	0.0447181	(-1.417692, -1.412148)
31	30	0	1	0	1	-1.41486	0.0287186	(-1.41664, -1.413080)
28	31	0	0	0	1	-1.41483	0.0695038	(-1.419138, -1.410522)
31	30	0	1	0	0	-1.41482	0.0351599	(-1.416999, -1.412641)
33	29	1	0	1	1	-1.41479	0.0323948	(-1.416798, -1.412782)
31	30	0	1	0	1	-1.41476	0.0771633	(-1.419543, -1.409977)
31	30	0	0	0	1	-1.41474	0.0271055	(-1.416420, -1.41306)
31	30	0	0	0	0	-1.41474	0.0296914	(-1.416580, -1.412900)
31	30	0	0	0	0	-1.41472	0.0537019	(-1.418048, -1.411392)
31	30	0	1	0	0	-1.41471	0.0303917	(-1.416594, -1.412826)
31	30	0	0	0	1	-1.41471	0.0290074	(-1.416508, -1.412912)
28	31	0	0	0	1	-1.41468	0.0272992	(-1.416372, -1.412988)
29	30	0	0	0	1	-1.41466	0.0515793	(-1.417857, -1.411463)
31	30	0	0	0	1	-1.41465	0.062767	(-1.418540, -1.410760)
31	30	0	0	0	1	-1.41465	0.033798	(-1.416745, -1.412555)
31	30	0	0	0	1	-1.41464	0.037005	(-1.416934, -1.412346)
31	30	0	0	0	1	-1.41464	0.0355932	(-1.416846, -1.412434)
31	30	0	1	0	1	-1.41459	0.0327626	(-1.416621, -1.412559)
31	30	0	0	0	0	-1.41457	0.0343976	(-1.416702, -1.412438)
31	30	0	0	0	1	-1.41456	0.034238	(-1.416682, -1.412438)
31	30	0	0	0	1	-1.41455	0.0281282	(-1.416293, -1.412807)
31	30	0	0	0	0	-1.41454	0.0259124	(-1.416146, -1.412934)
31	30	0	1	0	0	-1.41454	0.0724237	(-1.419029, -1.410051)

The performance of the second run of the GA is highlighted in Figures 4.5(a) and 4.5(b). The candidate allocation selected in this run suggests that 36 tests of Component 13, 27 tests of Component 33, and 9 tests of Component 14 should be performed. The candidates in the final generation of the second run were somewhat similar to those found in the first run

of the GA in that the majority of the resources were allocated to tests on Components 13 and 33. In addition, a non-trivial portion of the resources were used on Component 14 and other components were occasionally allocated a single test. The expected entropy of the best allocation was estimated to be -1.4153; a 95% confidence interval for the expected entropy of this candidate allocation, accounting for uncertainty associated with outcome sampling, is (-1.41846, -1.41214). The last allocation in the final generation had an estimated expected entropy of -1.41431; a 95% confidence interval for the expected entropy of this allocation is (-1.4162, -1.412419). Thus, there is considerable overlap in the final populations of the two GA runs when the sampling based evaluation procedure was used, and as a result, none of these allocations can be said to be clearly more informative than the others.



(a) Tests per component for genetic algorithm

(b) Estimated expected entropy for the best candidate allocation found in each generation

Figure 4.5 Performance of a second run of the genetic algorithm for finding an optimal allocation of resources in the missile case study under the first cost structure

Because there was no clear separation in the candidate allocations in these two final populations, we evaluate the 26 unique candidate allocations in the two final populations more precisely. These 26 allocations each call for testing only a few components, and thus the expected entropy can be computed without relying upon sampling (i.e., computed exactly within the simulation error of the MCMC) by enumerating all possible outcomes for the candidate allocations as in Section 2.3. The 26 unique candidate allocations, and their expected entropies, are displayed in Table 4.4. Because the candidates in the final population of the third GA run

were clearly inferior to those in the first two GA runs, they are not included here.

Table 4.4 Expected entropy for unique candidate allocations from the final populations of two runs of the GA

Flight Struct.	Warhead	Other Components									Expected
13	33	7	14	16	18	21	22	23	24	35	Entropy
36	28	1	1	1	0	0	0	0	0	0	-1.41242
31	30	0	0	0	0	0	1	0	1	0	-1.4124
31	30	0	0	0	0	0	1	0	0	0	-1.41238
33	28	0	9	0	0	0	0	0	0	0	-1.41238
31	30	0	0	0	0	0	0	0	1	0	-1.41236
31	30	0	0	0	0	0	0	0	0	0	-1.41234
34	29	0	0	0	0	0	0	0	0	0	-1.41234
37	27	1	5	0	0	0	1	0	0	0	-1.41231
36	27	0	9	0	0	0	0	0	0	0	-1.41229
30	29	0	8	1	0	0	0	0	0	0	-1.41227
28	31	0	0	0	0	0	0	0	1	0	-1.41225
28	31	0	0	0	0	0	0	0	0	0	-1.41223
30	29	0	8	0	0	0	0	0	0	0	-1.41222
33	28	1	6	0	0	0	0	0	0	0	-1.41219
40	26	0	7	1	0	0	0	0	0	0	-1.41217
36	27	0	8	0	0	0	0	0	0	0	-1.41214
27	30	0	8	0	0	0	0	0	0	0	-1.41211
33	29	0	0	0	0	1	0	1	1	0	-1.41209
42	26	0	3	0	0	0	0	1	0	0	-1.41208
33	27	0	14	0	0	1	0	1	1	0	-1.41207
37	27	1	3	1	1	1	0	0	0	0	-1.41207
31	28	0	11	0	0	0	0	1	1	1	-1.41205
35	27	0	8	0	0	0	0	0	0	0	-1.41182
37	27	0	3	0	0	0	0	1	0	0	-1.41171
29	30	0	0	0	0	0	0	0	1	0	-1.41164
33	27	0	9	0	0	0	0	0	0	0	-1.4113

The top-ranked allocation in Table 4.4, a candidate allocation identified in the second GA run, utilized most of the budget on tests of Components 13 and 33 (36 and 28 tests, respectively) and a single test on each of Components 7, 14, and 16. The expected entropy of this allocation was computed to be -1.41242. However, the expected entropies for these 26 candidate allocations are quite similar; the difference in expected entropy between the highest- and lowest-ranked allocations is only -0.0112. Thus, the expected entropies for the 26 candidates are so similar that, for all practical purposes, any of these allocations would be a good choice for collecting second-stage data.

In the first cost structure, several of the basic component testing costs were actually higher than the testing cost of the corresponding subsystem. This phenomenon may not be realistic, so we consider a second cost structure. In this cost structure, it is assumed that all components

in a subsystem have similar testing costs and no basic component's test costs more than a test on the corresponding subsystem. This second cost structure is displayed in Table 4.5. The budget for collected second-stage data is still restricted to be $5000.

Table 4.5 Second hypothetical testing costs for testable components in series missile system

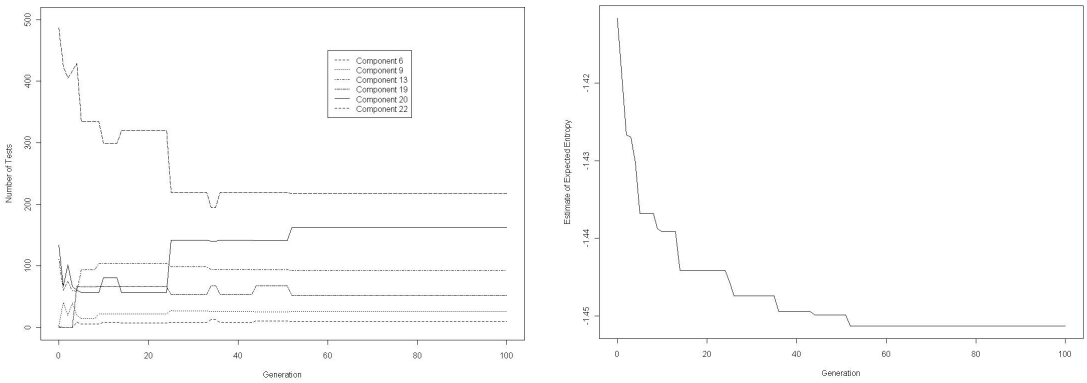
Component	Cost	Component	Cost	Component	Cost
Warhead (33)	$100	Missile (34)	$100	Aircraft (35)	$10
A (1)	$40	Power Supply (10)	$15	Flight Structure (16)	$5
B (2)	$40	Guidance Sys. (11)	$15	Avionics (17)	$5
C (3)	$40	Motor (12)	$15	Power (18)	$5
D (4)	$40	Flight Structure (13)	$15	Flight Control (19)	$5
E (5)	$40	Aircraft Interface (14)	$15	Environmental (20)	$5
F (6)	$40	Control (15)	$15	Acquisition (21)	$5
G (7)	$40			Launching (22)	$5
H (8)	$40			Missile Interface (23)	$5
I (9)	$40			Human Intervention (24)	$5

A genetic algorithm, with a population size of 30, was run for 100 generations to search for an optimal second-stage allocation. As before, this was repeated three times (i.e., three runs of the GA). The three runs selected quite different allocations, with the candidates in the final population for one run being considerably better than those in the other two runs; the best allocation found in this run is given in Table 4.6 and the performance of this GA run is displayed in Figures 4.6(a) and 4.6(b). The best allocation found by the GA includes tests of basic components in each of the testable subsystems. Components 6 and 9 in the warhead subsystem are allocated testing resources for collecting second-stage data; these two components had the fewest first-stage tests, with 10 and 5 tests performed, respectively. Tests were allocated to Component 13 (flight structure) in the missile subsystem; the flight structure in the missile subsystem had the fewest first-stage tests (25) and the lowest first-stage success rate (23/25) of all basic components in this subsystem. The remaining budget is allocated to tests of Components 19, 20, and 22 in the aircraft subsystem.

The 23 unique candidate allocations in the final population from the best GA run were re-evaluated by drawing samples of size 10,000. These allocations are too large to evaluate using

Table 4.6 Best allocation found by GA for second cost structure

Warhead Comps.		Missile Comps.	Aircraft Comps.			$\hat{G}$	$s_{\hat{G}}$
C_6	C_9	C_{13}	C_{19}	C_{20}	C_{22}		
10	26	93	218	52	162	-1.45131	0.15819



(a) Tests per component for genetic algorithm

(b) Estimated expected entropy for the best candidate allocation found in each generation

Figure 4.6 Performance of a single run of the genetic algorithm for finding an optimal allocation of resources under second cost structure in the missile case study

the methodology from Section 2.3. The estimated expected entropy, standard deviation and 95% normal-theory confidence intervals for the expected entropy for each candidate allocation are displayed in Table 4.7. The allocation identified by the GA was ranked 22^{nd} when the candidates were evaluated more precisely. The confidence interval for this allocation overlaps with that of the top-ranked allocation, indicating that one of these allocations cannot be said to be more informative than the other. The confidence interval for the expected entropy of the top-ranked allocation does not overlap with that for the lowest-ranked allocation, and thus the top-ranked allocation can be said to be more informative than the lowest-ranked allocation in the final population. The results of the GA suggest that most of the allocations in Table 4.7 would be a good choice for collecting second-stage data. Providing an experimenter with a list of good candidate allocations, such as those in Table 4.7, is benefitial from a practical perspective as some candidate experiments may be easier to perform than others.

Table 4.7 Final population in best GA for second cost structure for the missile case study

	Warhead		Missile	Aircraft						$\hat{G}$	$s_{\hat{G}}$	95% CI
	C_6	C_9	C_{13}	C_{16}	C_{17}	C_{18}	C_{19}	C_{20}	C_{22}			
1	10	28	94	0	0	0	183	68	163	-1.4418	0.123072	(-1.444212, -1.439388)
2	12	26	94	0	0	0	219	54	141	-1.44104	0.11055	(-1.443207, -1.438873)
3	10	31	119	0	1	1	158	65	89	-1.44021	0.107636	(-1.442320, -1.438100)
4	13	26	95	0	0	0	195	68	140	-1.44019	0.110497	(-1.442356, -1.438024)
5	12	21	93	0	0	0	217	97	141	-1.43991	0.120231	(-1.442267, -1.437553)
6	12	25	92	0	0	0	219	68	139	-1.43985	0.111057	(-1.442027, -1.437673)
7	12	21	115	1	0	0	165	97	112	-1.43965	0.110674	(-1.441819, -1.437481)
8	11	25	93	0	0	0	219	68	141	-1.43958	0.10711	(-1.441679, -1.437481)
9	10	26	93	0	0	0	216	52	162	-1.43909	0.113655	(-1.441318, -1.436862)
10	11	28	94	0	0	0	168	68	166	-1.43906	0.108167	(-1.441180, -1.436940)
11	12	27	115	1	0	0	165	64	112	-1.43903	0.113573	(-1.441256, -1.436804)
12	11	24	95	0	0	0	234	38	162	-1.43901	0.113669	(-1.441238, -1.436782)
13	11	25	94	0	0	0	219	68	141	-1.43884	0.114151	(-1.441077, -1.436603)
14	10	24	93	0	0	0	218	68	162	-1.43879	0.109843	(-1.440943, -1.436637)
15	11	25	95	0	0	0	219	68	139	-1.43878	0.107693	(-1.440891, -1.436669)
16	8	23	111	0	0	0	216	84	119	-1.43877	0.111329	(-1.440952, -1.436588)
17	11	25	93	0	0	0	218	51	162	-1.43848	0.11029	(-1.440642, -1.436318)
18	10	26	94	0	0	0	185	99	140	-1.43845	0.107694	(-1.440561, -1.436339)
19	10	26	95	0	0	0	219	68	139	-1.43822	0.107489	(-1.440327, -1.436113)
20	11	24	96	0	0	0	218	52	162	-1.43814	0.103131	(-1.440161, -1.436119)
21	8	26	95	1	0	0	219	54	142	-1.43795	0.112498	(-1.440155, -1.435745)
22	10	26	93	0	0	0	218	52	162	-1.43787	0.107666	(-1.439980, -1.435760)
23	10	25	94	0	0	0	219	52	141	-1.43675	0.0995918	(-1.438702, -1.434798)

The average computing time for these six GA runs (three for the first cost structure and three for the second) was 2.44 days with a standard deviation of 0.36 days. This suggests that there may be great computational advantages to using our candidate evaluation methodology with a genetic algorithm to find a good option for collecting second-stage data over the current approach.

Next, we consider finding an optimal resource allocation, under both cost structures, when the available budget is \$10,000. The GA, with population size 30 and 100 generations, was run three times for the first cost structure (described in Table 4.2). One run of the GA resulted in a final population that was considerably better than the other two. The best allocation found by this run of the GA allocated tests to components in each of the testable subsystems. Component 3 in the warhead subsystem was allocated 43 tests; Component 3 was the least expensive warhead component to have any first-stage failures. In the missile subsystem, 59 tests were allocated to Component 11 and 42 tests were allocated to Component 15. The remainder of the budget was allocated to 207 tests of Component 16, 101 tests of Component 18, 135 tests of Component 19, 85 tests of Component 20, 174 tests of Component 21, 138 tests of Component 22, 99 tests of Component 23, and 407 tests of Component 24 in the aircraft subsystem; all of these components were rather inexpensive to test. In addition, a single test of the aircraft subsystem (35) was included in the allocation found by the GA. The expected entropy of this candidate allocation was estimated to be -1.53426, with standard deviation 0.267176.

The 27 unique candidates in the final population of this run of the GA were evaluated more precisely by drawing 10,000 outcomes; the estimate of the expected entropy, the standard deviation, and a 95% confidence interval for the expected entropy of each candidate are displayed in Table 4.8. The candidate identified by the GA was ranked 5^{th} when the candidates were re-evaluated, but the confidence interval accounting for the sampling of outcomes overlaps with that of the top-ranked allocation, indicating that neither allocation is significantly more informative than the other. Further, the confidence interval for the top-ranked allocation in Table 4.8 only overlaps with those for candidates 2 - 19; thus, this allocation is significantly

Table 4.8 Final population in best GA for first cost structure for the missile case study with a budget of $10,000; the allocations are represented as the number of tests of each of the following components: (C_3, C_5, C_{11}, C_{15}, C_{16}, C_{17}, C_{18}, C_{19}, C_{20}, C_{21}, C_{22}, C_{23}, C_{24}, C_{35})

	Allocation	$\hat{G}$	$s_{\hat{G}}$	95% CI
1	(41, 0, 52, 42, 156, 0, 150, 176, 64, 199, 137, 97, 404, 0)	-1.51231	0.23466	(-1.51691, -1.50771)
2	(34, 0, 36, 29, 167, 0, 158, 180, 89, 209, 168, 86, 459, 0)	-1.51125	0.22322	(-1.51563, -1.50688)
3	(36, 0, 37, 29, 148, 0, 192, 164, 91, 209, 157, 88, 460, 1)	-1.50991	0.22591	(-1.51434, -1.50548)
4	(39, 1, 54, 28, 166, 1, 189, 142, 89, 134, 196, 88, 349, 2)	-1.50966	0.22907	(-1.51415, -1.50517)
5	(43, 0, 59, 42, 207, 0, 101, 135, 85, 174, 138, 99, 407, 1)	-1.5095	0.23229	(-1.51405, -1.50495)
6	(43, 0, 51, 37, 131, 1, 142, 157, 112, 157, 198, 93, 343, 1)	-1.50919	0.21096	(-1.51333, -1.50506)
7	(36, 0, 52, 34, 160, 0, 128, 180, 111, 150, 168, 78, 427, 0)	-1.50897	0.23019	(-1.51348, -1.50446)
8	(25, 0, 71, 29, 189, 0, 178, 181, 61, 222, 132, 92, 329, 0)	-1.50865	0.22780	(-1.51312, -1.50419)
9	(27, 0, 62, 35, 162, 0, 153, 180, 67, 204, 167, 85, 397, 0)	-1.50839	0.22054	(-1.51271, -1.50407)
10	(34, 0, 40, 36, 184, 0, 168, 112, 89, 240, 176, 78, 436, 0)	-1.5083	0.22783	(-1.51277, -1.50384)
11	(25, 0, 62, 32, 160, 0, 157, 178, 87, 202, 165, 82, 396, 0)	-1.50738	0.22830	(-1.51186, -1.50291)
12	(36, 0, 37, 23, 158, 0, 128, 208, 102, 147, 171, 90, 543, 0)	-1.50724	0.23137	(-1.51178, -1.50271)
13	(34, 0, 37, 33, 162, 0, 158, 178, 88, 238, 165, 76, 434, 0)	-1.50684	0.22121	(-1.51118, -1.50250)
14	(36, 0, 63, 23, 167, 0, 159, 180, 67, 150, 168, 88, 400, 0)	-1.50647	0.22495	(-1.51088, -1.50206)
15	(35, 0, 38, 36, 180, 0, 163, 112, 84, 234, 176, 78, 432, 0)	-1.50599	0.21728	(-1.51025, -1.50173)
16	(36, 0, 37, 35, 148, 0, 191, 179, 91, 149 , 167, 76, 460, 1)	-1.50572	0.21873	(-1.51001, -1.50143)
17	(31, 0, 35, 25, 195, 0, 123, 203, 78, 234, 201, 61, 431, 0)	-1.50445	0.22192	(-1.50880, -1.50010)
18	(30, 0, 56, 35, 162, 0, 153, 176, 70, 204, 164, 75, 431, 0)	-1.50421	0.21423	(-1.50841, -1.50001)
19	(36, 0, 36, 36, 167, 0, 159, 180, 89, 150, 168, 78, 459, 0)	-1.50355	0.21514	(-1.50777, -1.49933)
20	(45, 0, 52, 34, 156, 0, 132, 157, 114, 150, 160, 78, 426, 0)	-1.50352	0.20261	(-1.50749, -1.49955)
21	(36, 0, 40, 36, 167, 0, 159, 180, 89, 150, 168, 78, 436, 0)	-1.50299	0.20736	(-1.50705, -1.49893)
22	(34, 0, 60, 26, 186, 0, 158, 191, 75, 207, 164, 66, 321, 0)	-1.50287	0.21349	(-1.50705, -1.49869)
23	(36, 0, 36, 36, 148, 0, 192, 179, 90, 150, 168, 77, 459, 1)	-1.50271	0.20982	(-1.50682, -1.49860)
24	(36, 0, 36, 36, 167, 0, 159, 178, 89, 150, 165, 78, 459, 0)	-1.5023	0.21886	(-1.50659, -1.49801)
25	(37, 0, 63, 17, 188, 0, 159, 195, 76, 161, 168, 68, 323, 0)	-1.50228	0.22032	(-1.50660, -1.49796)
26	(34, 0, 57, 40, 207, 0, 127, 134, 110, 173, 136, 96, 405, 0)	-1.50158	0.20502	(-1.50560, -1.49756)
27	(28, 0, 41, 28, 158, 1, 132, 210, 102, 119, 175, 93, 545, 1)	-1.50084	0.21765	(-1.50511, -1.49657)

more informative than allocations 20 - 27.

In addition, the top-ranked allocation in Table 4.4 (i.e., that identified as being best under the first cost structure for a budget of $5000) was doubled and evaluated by drawing 10,000 outcomes. The 95% confidence interval for the expected entropy of this allocation was computed to be (-1.445578, -1.443602); there is no overlap between this confidence interval and any of those for the candidates in Table 4.8, indicating that this allocation is not as informative as those in Table 4.8.

As a check, the candidate allocation identified by the GA in this scenario was halved (in order to satisfy a budget of $5000) and evaluated by drawing a sample of size 10,000. A 95% confidence interval for the expected entropy of this allocation was found to be (-1.400364, -1.397096), which is considerably less informative than the allocations in Table 4.4. This suggests that there exists a point of diminishing returns; that is, at some point, using additional resources to collect more tests on Component 13 and 33 is not as informative as using those resources for tests of other components.

Under the second cost structure (described in Table 4.5), a genetic algorithm was implemented to search for an optimal resource allocation with a budget of $10,000. As before, the GA with a population size of 30 and 100 generations was run 3 times. The best allocation found once again consisted of tests of components in each of the testable subsystems. In the warhead subsystem, 7 tests were allocated to Component 3 and 33 tests to Component 9. Components 11 and 13 in the missile subsystem were allocated 57 and 91 tests, respectively. The best allocation also included 196 tests of Component 16, 252 tests of Component 17, 343 tests of Component 18, 269 tests of Component 19, 173 tests of Component 20, and 2 tests of Component 23.

The 24 unique candidates were re-evaluated using samples of size 10,000; the expected entropy estimate, standard deviation, and a 95% confidence interval, accounting for variability due to sampling outcomes, are displayed in Table 4.9. The allocation identified as best by the GA was ranked 8^{th} when the candidates were evaluated more precisely. The confidence interval for this allocation overlaps with that for the top-ranked allocation in Table 4.9, indicating that

Table 4.9 Final population in best GA for second cost structure for the missile case study with a budget of $10,000; allocations represent the number of tests on the following components: (C_3, C_9, C_{11}, C_{13}, C_{16}, C_{17}, C_{18}, C_{19}, C_{20}, C_{21}, C_{22}, C_{23}, C_{24}, C_{35})

	Allocation	$\hat{G}$	$s_{\hat{G}}$	95% CI
1	(9, 27, 59, 94, 198, 256, 351, 271, 174, 0, 0, 0, 0, 0)	-1.6142	0.328496	(-1.620639, -1.607761)
2	(0, 31, 63, 109, 268, 180, 282, 279, 223, 2, 0, 1, 1, 0)	-1.60509	0.305765	(-1.611083, -1.599097)
3	(4, 33, 59, 92, 198, 256, 352, 269, 171, 0, 0, 3, 0, 1)	-1.60389	0.319489	(-1.610152, -1.597628)
4	(7, 33, 54, 92, 198, 256, 343, 269, 173, 0, 0, 3, 0, 0)	-1.60362	0.319027	(-1.609873, -1.597367)
5	(6, 33, 57, 92, 198, 252, 343, 269, 173, 0, 0, 1, 0, 0)	-1.60346	0.317448	(-1.609682, -1.597238)
6	(7, 33, 56, 93, 195, 250, 349, 268, 168, 0, 0, 3, 0, 0)	-1.60319	0.311896	(-1.609303, -1.597077)
7	(5, 33, 59, 91, 198, 256, 351, 269, 167, 0, 0, 4, 0, 0)	-1.60274	0.324301	(-1.609096, -1.596384)
8	(7, 33, 57, 91, 196, 252, 343, 269, 173, 0, 0, 2, 0, 0)	-1.60246	0.313125	(-1.608597, -1.596323)
9	(7, 33, 55, 93, 196, 252, 349, 268, 168, 0, 0, 2, 0, 0)	-1.60171	0.306813	(-1.607724, -1.595696)
10	(4, 33, 57, 93, 198, 256, 352, 268, 171, 0, 0, 3, 0, 1)	-1.60157	0.315135	(-1.607747, -1.595393)
11	(7, 32, 57, 91, 196, 255, 351, 269, 173, 0, 0, 0, 0, 0)	-1.60116	0.314899	(-1.607332, -1.594988)
12	(11, 33, 57, 86, 178, 245, 369, 228, 184, 2, 2, 1, 2, 1)	-1.6009	0.305171	(-1.606881, -1.594919)
13	(5, 33, 58, 92, 198, 256, 351, 269, 167, 0, 0, 4, 0, 0)	-1.60053	0.313901	(-1.606682, -1.594378)
14	(5, 33, 60, 96, 196, 252, 342, 269, 167, 0, 0, 0, 0, 0)	-1.60052	0.29825	(-1.606366, -1.594674)
15	(3, 33, 57, 97, 197, 254, 274, 312, 209, 0, 0, 4, 0, 0)	-1.60047	0.308225	(-1.606511, -1.594429)
16	(7, 33, 57, 91, 196, 252, 343, 269, 173, 0, 0, 0, 0, 0)	-1.59965	0.30363	(-1.605601, -1.593699)
17	(4, 33, 57, 91, 198, 252, 343, 269, 171, 0, 0, 3, 0, 1)	-1.59885	0.305279	(-1.604833, -1.592867)
18	(6, 33, 59, 91, 198, 256, 343, 269, 166, 0, 0, 4, 0 , 0)	-1.59885	0.307921	(-1.604885, -1.592815)
19	(7, 33, 58, 91, 197, 256, 343, 268, 167, 0, 0, 2, 0, 0)	-1.59878	0.301789	(-1.604695, -1.592865)
20	(9, 27, 57, 91, 198, 256, 343, 271, 174, 0, 0, 2, 0, 0)	-1.59804	0.305289	(-1.604024, -1.592056)
21	(5, 31, 54, 101, 202, 279, 310, 278, 152, 2, 1, 2, 1, 0)	-1.59645	0.29841	(-1.602299, -1.590601)
22	(9, 27, 53, 94, 199, 261, 352, 271, 174, 0, 0, 0, 0, 0)	-1.59396	0.292798	(-1.599699, -1.588221)
23	(7, 33, 56, 91, 198, 252, 343, 268, 168, 0, 0, 3, 0, 1)	-1.59388	0.294914	(-1.599660, -1.588100)
24	(7, 27, 57, 94, 198, 252, 351, 271, 173, 0, 0, 0, 0, 0)	-1.59313	0.291539	(-1.598844, -1.587416)

one cannot be said to be more informative than the other. However, the confidence interval for the top-ranked allocation does not overlap with those for allocations 9 - 24, indicating that the top-ranked allocation is significantly more informative than these allocations.

We also consider doubling the top-ranked allocation in Table 4.7 (i.e., the top-ranked allocation under the second cost structure when the budget is \$5000); a 95% confidence interval for the expected entropy of this allocation is (-1.54014, -1.531126). Thus simply doubling the top-ranked allocation for a budget of \$5000 does not yield the most informative allocation for a \$10,000 budget. In this example, halving the top-ranked allocation in Table 4.9 (to satisfy a \$5000 budget) and evaluating the allocation by sampling 10,000 outcomes yields the 95% confidence interval (-1.445395, -1.440485). This confidence interval overlaps with the confidence interval for the top-ranked allocation in Table 4.7, indicating that one of the allocations cannot be said to be more informative than the other.

Once again, the average computing time was quite fast. These six GA runs (three for the first cost structure and three for the second) had an average computing time of 3.32 days, with standard deviation 0.12 days. This provides further evidence of the computational efficiency of our approach.

In this case study, we considered finding an optimal allocation of resources for the missile system described by Martz *et al* (1988) under different hypothetical cost structures and budgets. For a given cost structure and budget, many allocations of similar structure were found to be nearly optimal. However, changing the cost structure greatly changed the structure of the allocations found by the GAs. Additionally, we found that doubling the available budget for each cost structure did not necessarily equate to doubling the best allocations. Lastly, the GAs implemented in this section required a fairly small amount of computing time, indicating that our approach is quite computationally efficient and could be considered as a feasible alternative to the current approach.

4.2 Demand Unavailability of the Low-Pressure Coolant Injection System in Nuclear-Power Boiling-Water Reactors

Martz and Waller (1990) discuss an application in which they estimate the demand unavailability of one of the safety features in a certain 1,150 megawatt electric U.S. commercial nuclear-power boiling-water reactor. In this reactor, the low-pressure coolant injection (LPCI) system provides coolant to the reactor vessel during accidents in which the vessel pressure is low (Martz and Waller (1990)). The LPCI system usually operates on standby, waiting for a usage demand; thus the components in this system must perform on demand (Martz and Waller (1990)). Failures of the system and the components that comprise the system may be due to either failure to start on demand or unscheduled maintenance; Martz and Waller (1990) modeled only failure to start on demand and we shall do the same. Figure 4.7 displays the demand availability block diagram for the LPCI system. The goal of the study was to estimate the demand unavailability (failure to start while operating on standby) based on current binomial data and historical data on the basic components and historical data only on the major subsystems.

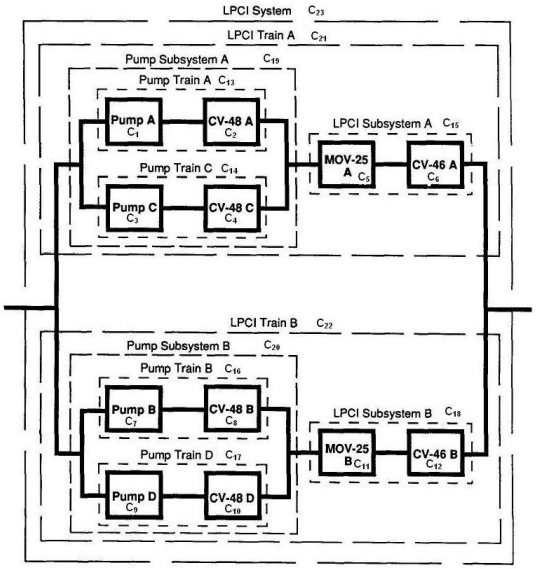


Figure 4.7 LPCI system demand availability block diagram (Figure 1 from Martz and Waller (1990))

In this system, current binomial data are available for all basic components, but not for any higher-level components. Martz and Waller (1990) note that there are redundancies in this system, but that each component is assumed to have its own underlying demand availability value. The current binomial data and historical data for the LPCI system are provided in Table 4.10. Martz and Waller (1990) perform an analysis similar to the two-stage Bayesian analysis described in Section 4.1; the LPCI system has more levels than the system described in Section 4.1, adding more steps to the analysis. As before, their analysis is outlined here, with additional details provided in Appendix C. They first consider subsystems comprised only of basic components.

1. For each basic component in the system, the historical data for that basic component are used to form a prior distribution for the reliability of that basic component. The current binomial test data for that basic component are used to update the basic component's reliability prior distribution to obtain the corresponding basic component reliability posterior distribution.

2. Suppose Subsystem i is comprised of k_i basic components in series. The Subsystem i reliability induced prior distribution is the product of the k_i basic component reliability posterior distributions.

3. The Subsystem i reliability native prior distribution is formed from the Subsystem i historical data.

4. The Subsystem i reliability induced and native prior distributions yield the Subsystem i combined prior distribution (i.e., the Subsystem i reliability combined prior distribution uses the historical data for Subsystem i and its basic components *and* the current binomial test data for the basic components in Subsystem i).

5. The current binomial test data for Subsystem i are used to update the Subsystem i reliability combined prior distribution to yield the Subsystem i reliability posterior distribution. (Steps 2 - 5 are repeated for all subsystems comprised of basic components.)

They next consider subsystems comprised of other subsystems and derive the corresponding subsystem reliability posterior distributions in a manner similar to that outlined above. After the subsystem reliability posterior distributions have been derived for all subsystems, the system-level analysis again follows from the subsystem-level analysis.

Table 4.10 First-stage data and expert best guesses for LPCI system (Martz and Waller (1990))

Component		Type	Data	π
Pump A	1	Basic	236/240	191.17/191.79
CV-48 A	2	Basic	240/240	14232.34/14234.12
Pump C	3	Basic	238/240	191.17/191.79
CV-48 C	4	Basic	240/240	14232.34/14234.12
MOV-25 A	5	Basic	240/240	470.13/471.90
CV-46 A	6	Basic	240/240	14232.34/14234.12
Pump B	7	Basic	240/240	191.79/191.79
CV-48 B	8	Basic	240/240	14232.34/14234.12
Pump D	9	Basic	238/240	191.17/191.79
CV-48 D	10	Basic	240/240	14232.34/14234.12
MOV-25 B	11	Basic	240/240	470.13/471.90
CV-46 B	12	Basic	240/240	14232.34/14234.12
Pump Train A	13	Subsystem		1.55/1.58
Pump Train C	14	Subsystem		1.55/1.58
LPCI Subsystem A	15	Subsystem		242.87/244.66
Pump Train B	16	Subsystem		1.55/1.58
Pump Train D	17	Subsystem		1.55/1.58
LPCI Subsystem B	18	Subsystem		242.87/244.66
Pump Subsystem A	19	Subsystem		
Pump Subsystem B	20	Subsystem		
LPCI Train A	21	Subsystem		
LPCI Train B	22	Subsystem		
LPCI System	23	System		

In order to address the resource allocation problem, we first analyze these data using the model described in Section 1.2. The complete details of the analysis are provided in Appendix C. The system and subsystem combined prior distributions used by Martz and Waller (1990) (dashed lines) and those specified in the model of Johnson *et al* (2003) (solid lines) are displayed in Figure 4.8. As in the previous case study, the system and subsystem priors of Martz and Waller (1990) tend to be considerably more informative than those used in the model of Johnson *et al* (2003).

The system and subsystem reliability posterior distributions from the model of Martz and Waller (1990) (dashed) and that of Johnson *et al* (2003) (solid) are displayed in Figure 4.9. Because there were only current binomial data available at the basic component level, the

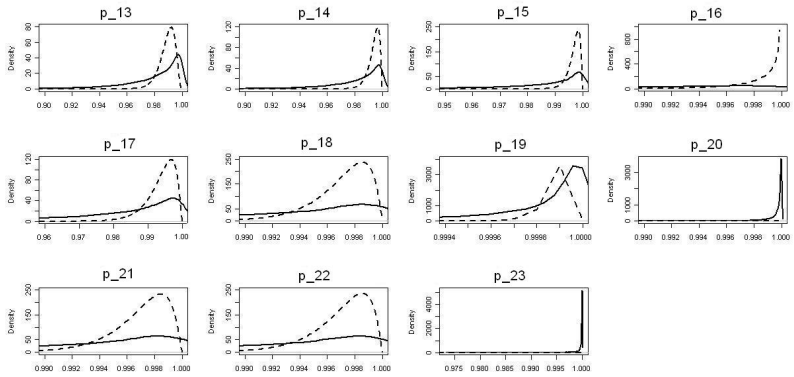


Figure 4.8 Prior distributions for the reliability of subsystems and LPCI system from Johnson *et al* (2003) model (solid) and Martz and Waller (1990) model (dashed)

system and subsystem posterior distributions of Martz and Waller (1990) are the same as the combined prior distributions (i.e., the basic component binomial data were used to create the subsystem and system combined prior distributions). Using the model of Johnson *et al* (2003), the LPCI system demand availability posterior mean is estimated to be 0.999997. Thus an estimate of the system demand unavailability, the purpose of the initial study, is 3×10^{-6}; that is, we might expect, on average, the LPCI system to fail roughly once in every 333,300 usage demands. The standard deviation of the system availability (and unavailability) posterior distribution is 8.31×10^{-6}.

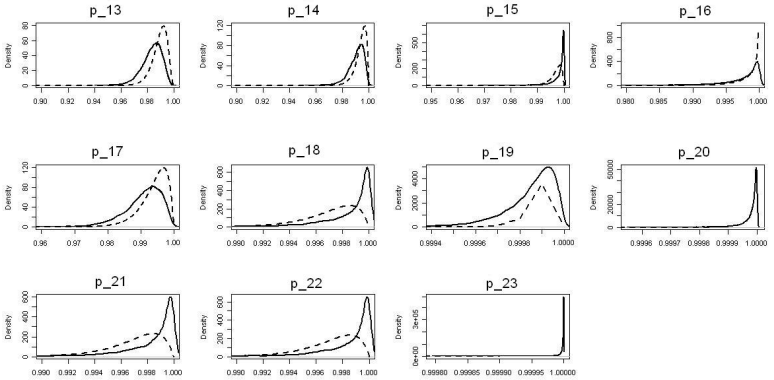


Figure 4.9 Posterior distributions for the reliability of subsystems and LPCI system of Johnson *et al* (2003) model (solid) and Martz and Waller (1990) model (dashed)

In Section 2.5, we suggest that the bin width chosen to calculate entropy should be a small fraction of the posterior standard deviation to reduce the amount of numerical error associated with the entropy calculation; also in Section 2.5, and everywhere the histogram approximation is used to compute the entropy of the updated posterior distribution, the histogram was constructed over the entire interval (0, 1) and thus the corresponding number of bins used was $n_h = \lceil \frac{1}{h} \rceil$. For all systems considered thus far, this approach to constructing the histogram had not caused any problems. However, in the LPCI case study, the draws from the system demand availability posterior distribution cover only a small region of the interval (0, 1). Therefore, to calculate the entropy of the posterior distribution, and perform resource allocation, we calculate the histogram over only part of the interval (0, 1); specifically, we restrict our attention to the interval (0.9997945, 1), which is an interval slightly wider than the range of the LPCI system demand availability posterior draws.

Because the reliability of this system is so much more extreme than any other system discussed previously, we considered using several different bin widths to compute the entropy of the LPCI system demand availability posterior distribution and the expected entropy for candidate allocations: $h_1 \approx \frac{\hat{\sigma}}{6}$, $h_2 \approx \frac{\hat{\sigma}}{10}$, and $h_3 \approx \frac{\hat{\sigma}}{16}$. Based on the 10,000 posterior draws, the system reliability posterior entropy calculated using these bin widths is -11.9818, -12.1041, and -12.1904, respectively.

To address resource allocation, we again assume that the only testable components are those for which there was initial data; thus the basic components are the only testable components in the LPCI system. We assume that tests on the basic components cost \$4 each and consider a second-stage budget of \$2000, which is roughly $\frac{1}{6}$ of the implied first-stage budget. A genetic algorithm, with population size 30, was run for 100 generations to search for an optimal resource allocation; the candidate allocations in the GA were evaluated by drawing samples of size 1000 and computing an estimate of the expected entropy as in Section 2.6. For each bin width, the GA was run three times.

We first consider the bin width $h \approx \frac{\hat{\sigma}}{6}$. The three runs ended with slightly different final populations, though all of these candidates use the available resources to test some combination

Table 4.11 Best allocation in each of the three runs of GA using $h \approx \frac{\hat{\sigma}}{6}$

	$(C_5, C_6, C_{11}, C_{12})$	$\hat{G}$	$s_{\hat{G}}$	95% CI
Run 1	(187, 66, 193, 54)	-12.3102	0.610739	(-12.34805, -12.27235)
Run 2	(132, 113, 251, 0)	-12.2885	0.606539	(-12.32609, -12.25091)
Run 3	(95, 21, 251, 133)	-12.308	0.647008	(-12.3481, -12.2679)

of Components 5, 6, 11, and 12 (motor-operated valve-25 A, check valve-46, motor-operated valve-25 B, and check valve-46 B). The best allocation found in each run and a 95% confidence interval for the expected entropy for each allocation, to account for variability due to sampling outcomes, are displayed in Table 4.11. The confidence intervals for these three allocations overlap, so we cannot conclude that one is more informative than the others.

All candidates in the final population of the three GAs consisted of some combination of Components 5, 6, 11, and 12. The 58 unique candidate allocations from the final populations of the three runs were re-evaluated using samples of size 10,000. The estimate of the expected entropy for each candidate, the standard deviation, and a 95% confidence interval for the expected entropy for each candidate allocation are provided in Table 4.12. The confidence intervals for each pair of candidate allocations overlap, suggesting that, even when using a larger sample size to evaluate the candidates, we cannot conclude that one of these allocations is more informative the others.

Based on the first-stage data alone, these candidate allocations may be slightly surprising. In the initial experiment only three components saw any failures (Components 1, 3, and 9, i.e., Pumps A, C, and D, respectively). As a result, we might have expected the optimal resource allocation to include tests of these three components. After the genetic algorithm was implemented and the unique candidates in the final populations were re-evaluated, three additional "obvious" candidate allocations were considered: one of these candidates split the resources evenly between Components 1, 3, and 9; the second allocation used half the budget on Component 1 (which saw 2 failures in the initial experiment) and a fourth of the budget on each of Component 3 and 9 (both of which had a single failure in the initial experiment); and the last allocation contained tests on all basic components (41 tests on each of CV-48 A, B, C, and D and 42 tests on each of the remaining basic components). An estimate of the expected

Table 4.12 Estimate of the expected entropy of the unique candidate allocations from the final populations of the three runs of GA using $h \approx \frac{\hat{\sigma}}{6}$ and sample size N=10,000 to evaluate allocations

Allocation (C_5, C_6, C_{11}, C_{12})	$\hat{G}$	$s_{\hat{G}}$	95% CI	Allocation (C_5, C_6,C_{11}, C_{12})	$\hat{G}$	$s_{\hat{G}}$	95% CI
(187, 66, 193, 54)	-12.0932	0.633376	(-12.105614, -12.080786)	(145, 126, 229, 0)	-12.0791	0.61649	(-12.091183, -12.067017)
(176, 70, 217, 37)	-12.0846	0.635042	(-12.097047, -12.072153)	(107, 145, 248, 0)	-12.0822	0.609122	(-12.094139, -12.070261)
(159, 70, 231, 40)	-12.089	0.625682	(-12.101263, -12.076737)	(96, 177, 227, 0)	-12.0796	0.594631	(-12.091255, -12.067945)
(129, 49,322, 0)	-12.0837	0.599187	(-12.095444, -12.071956)	(123, 113, 254, 0)	-12.0893	0.598636	(-12.101033, -12.077567)
(160, 65, 214, 42)	-12.0864	0.616727	(-12.098488, -12.074312)	(95, 21, 251, 133)	-12.0856	0.643429	(-12.098211, -12.072989)
(167, 0, 233, 66)	-12.0893	0.608066	(-12.101218, -12.077382)	(118, 37, 188, 157)	-12.092	0.635823	(-12.104462, -12.079538)
(184, 26,231, 59)	-12.0822	0.638287	(-12.094710, -12.069690)	(185, 0, 212, 53)	-12.078	0.604554	(-12.089849, -12.066151)
(178, 56, 221, 45)	-12.0904	0.62593	(-12.102668, -12.078132)	(170, 0, 184, 146)	-12.0757	0.644107	(-12.088324, -12.063076)
(178, 99, 222, 1)	-12.0812	0.616091	(-12.093275, -12.069125)	(171, 8, 200, 121)	-12.1026	0.625462	(-12.114859, -12.090341)
(194, 0, 253, 53)	-12.0858	0.61891	(-12.097931, -12.073669)	(140, 58, 184, 118)	-12.0786	0.646149	(-12.091265, -12.065935)
(194, 63, 186, 57)	-12.0939	0.623839	(-12.106127, -12.081673)	(117, 52, 331, 0)	-12.0699	0.606138	(-12.081780, -12.058020)
(176, 66, 217, 37)	-12.071	0.629392	(-12.083336, -12.058664)	(82, 9, 309, 100)	-12.0899	0.627871	(-12.102206, -12.077594)
(187, 60, 192, 54)	-12.0943	0.620304	(-12.106458, -12.082142)	(115, 23, 181, 181)	-12.0841	0.640702	(-12.096658, -12.071542)
(227, 49, 174, 50)	-12.0958	0.616735	(-12.107888, -12.083712)	(112, 49, 215, 124)	-12.0834	0.644568	(-12.096034, -12.070766)
(184, 47, 232, 37)	-12.0873	0.624061	(-12.099532, -12.075068)	(178, 0, 299, 23)	-12.091	0.597761	(-12.102716, -12.079284)
(166, 63, 205, 66)	-12.0932	0.629413	(-12.105536, -12.080864)	(28, 25, 269, 120)	-12.0574	0.618614	(-12.069525, -12.045275)
(132, 113, 251, 0)	-12.0773	0.605139	(-12.089161, -12.065439)	(176, 1, 275, 48)	-12.0753	0.619799	(-12.087448, -12.063152)
(127, 148, 225, 0)	-12.0914	0.598597	(-12.103133, -12.079667)	(51, 0, 328, 121)	-12.0847	0.622205	(-12.096895, -12.072505)
(123, 116, 254, 0)	-12.0695	0.616995	(-12.081593, -12.057407)	(126, 60, 184, 130)	-12.0859	0.642764	(-12.098498, -12.073302)
(127, 138, 230, 0)	-12.0909	0.596668	(-12.102595, -12.079205)	(84, 36, 243, 137)	-12.0859	0.641358	(-12.098471, -12.073329)
(102, 167, 215, 0)	-12.0748	0.601984	(-12.086599, -12.063001)	(86, 26, 245, 143)	-12.0883	0.636184	(-12.100769, -12.075831)
(139, 112, 249, 0)	-12.0802	0.615081	(-12.092256, -12.068144)	(176, 21, 251, 48)	-12.0893	0.616761	(-12.101389, -12.077211)
(107, 154, 220, 0)	-12.0746	0.599355	(-12.086347, -12.062853)	(139, 0, 267, 94)	-12.0823	0.629604	(-12.094640, -12.069960)
(132, 148, 207, 0)	-12.0788	0.604344	(-12.090645, -12.066955)	(127, 16, 193, 164)	-12.0909	0.635498	(-12.103356, -12.078444)
(143, 135, 222, 0)	-12.0813	0.615953	(-12.093373, -12.069227)	(197, 0, 269, 34)	-12.0855	0.612024	(-12.097496, -12.073504)
(132, 138, 230, 0)	-12.078	0.607468	(-12.089906, -12.066094)	(176, 1, 275, 48)	-12.079	0.627682	(-12.091303, -12.066697)
(148, 128, 224, 0)	-12.0899	0.604973	(-12.101757, -12.078043)	(82, 0, 309, 100)	-12.0792	0.626394	(-12.091477, -12.066923)
(127, 116, 254, 0)	-12.0823	0.6101	(-12.094258, -12.070342)	(75, 32, 301, 92)	-12.0908	0.626866	(-12.103087, -12.078513)
(141, 133, 226, 0)	-12.0763	0.61291	(-12.088313, -12.064287)	(112, 0, 222, 166)	-12.0822	0.638448	(-12.094714, -12.069686)

Table 4.13 Candidate allocations that use the available resources to collect more tests on the basic components that had first-stage failures

Allocation (C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9, C_{10}, C_{11}, C_{12})	$\hat{G}$	$s_{\hat{G}}$	95% CI
(167, 0, 166, 0, 0, 0, 0, 0, 167, 0, 0, 0)	-11.9892	0.0551554	(-11.990281, -11.988119)
(250, 0, 125, 0, 0, 0, 0, 0, 125, 0, 0, 0)	-11.988	0.0568858	(-11.989115, -11.986885)
(42, 41, 42, 41, 42, 42, 42, 41, 42, 41, 42, 42)	-12.026	0.41066	(-12.03405, -12.01795)

entropy (based on samples size of 10,000), the standard deviation, and a 95% confidence interval for the expected entropy for each candidate, to account for sampling outcomes, are displayed in Table 4.13. The confidence intervals for the expected entropy of these allocations do not overlap with those for the unique candidates in the final populations of the three GA runs; we can conclude that these three allocations are not as informative as those found by the genetic algorithm.

When we re-examine the LPCI system, the allocations chosen by the genetic algorithms are reasonable choices, as Components 5, 6, 10, and 11 could be considered to be more "system-critical" than the other basic components. That is, if either Component 5 (MOV-25 A) or Component 6 (CV-48 A) fail, the LPCI Subsystem A and hence the LPCI Train A will fail. However, if Component 1 (Pump A) fails, either Component 3 (Pump C) or Component 4 (CV-46 C), or both, must fail in order for Pump Subsystem A and hence the LPCI Train A to fail. Similar reasoning holds for the LPCI Train B. The allocations chosen by the multiple runs of the genetic algorithm should yield more information about these system-critical components, and hence improve the information available about the reliability the LPCI system.

Two additional bin widths, $h_2 \approx \frac{\hat{\sigma}}{10}$ and $h_3 \approx \frac{\hat{\sigma}}{16}$, were considered because of the anticipated high amount of numerical error in the entropy calculations. For each of these additional bin widths, the genetic algorithm was run three times. The best allocation from each run for $h_2 \approx \frac{\hat{\sigma}}{10}$ is displayed in Table 4.14 and those for $h_3 \approx \frac{\hat{\sigma}}{16}$ are displayed in Table 4.15. In both cases, the best allocations found were similar to those when the bin width $h_1 \approx \frac{\hat{\sigma}}{6}$ was used; that is, the best allocations contain tests of some combination of Components 5, 6, 11, and 12. Also, in each case, the confidence intervals for the expected entropy of the allocation found by the GA in each run overlap, indicating that we cannot conclusively conclude which of the

Table 4.14 Best allocation in each of the three runs of GA using $h \approx \frac{\hat{\sigma}}{10}$

	$(C_5, C_6, C_{11}, C_{12})$	$\hat{G}$	$s_{\hat{G}}$	95% CI
Run 1	(89, 73, 222, 116)	-12.2971	0.638183	(-12.33556, -12.25755)
Run 2	(111, 88, 176, 116)	-12.2954	0.632042	(-12.33457, -12.25623)
Run 3	(0, 0, 203, 297)	-12.276	0.599613	(-12.31316, -12.23884)

Table 4.15 Best allocation in each of the three runs of GA using $h \approx \frac{\hat{\sigma}}{16}$

	$(C_5, C_6, C_{11}, C_{12})$	$\hat{G}$	$s_{\hat{G}}$	95% CI
Run 1	(0, 0, 272, 226)	-12.3882	0.645852	(-12.42823, -12.34817)
Run 2	(53, 42, 280, 125)	-12.3923	0.665068	(-12.43352, -12.35108)
Run 3	(158, 0, 174, 168)	-12.3954	0.665932	(-12.43667, -12.35413)

three allocations (for a specified bin width) is the most informative.

In both cases ($h_2 \approx \frac{\hat{\sigma}}{10}$ and $h_3 \approx \frac{\hat{\sigma}}{16}$), as with $h_1 \approx \frac{\hat{\sigma}}{6}$, the final populations for the three runs were re-evaluated using samples of size 10,000 and each pair of 95% confidence intervals for the expected entropy of a candidate in the final populations overlapped, indicating that we cannot conclusively claim that one candidate in the final population is more informative than another. Additionally, the three candidates described in Table 4.13 were evaluated using samples of size 10,000 and the bin widths $h_2 \approx \frac{\hat{\sigma}}{10}$ and $h_3 \approx \frac{\hat{\sigma}}{16}$; in both cases, these potential candidates were found to be significantly less informative than those found by the genetic algorithms. Because the three different bin widths yielded very similar best allocations, we can feel comfortable that even though there is numerical error in the entropy calculation, it does not seem to impact the relative rankings of the calculated entropies.

The nine allocations identified by the three GA runs using the three bin widths are displayed in Table 4.16. These nine allocations were each re-evaluated using the three bin widths and samples of size 10,000; the estimated expected entropy and a 95% confidence interval for the expected entropy for each candidate allocation, using all bin widths, are also provided. The two candidates that utilize all resources for tests on Components 11 and 12 were ranked as the least informative using all three bin widths. The remaining seven allocations are ranked differently for the three bin widths, but for a specific bin width the confidence intervals for these allocations overlap, indicating that one of these seven allocations cannot be said to be significantly more informative than another.

Table 4.16 Best allocations found by the three GA runs using bin widths $h_1 \approx \frac{\hat{\sigma}}{6}$, $h_2 \approx \frac{\hat{\sigma}}{10}$, and $h_3 \approx \frac{\hat{\sigma}}{16}$, each re-evaluated using samples of size $N = 10,000$ and all three bin widths

$(C_5, C_6, C_{11}, C_{12})$	$h_1 \approx \frac{\hat{\sigma}}{6}$		$h_2 \approx \frac{\hat{\sigma}}{10}$		$h_3 \approx \frac{\hat{\sigma}}{16}$	
	$\hat{G}$	95% CI	$\hat{G}$	95% CI	$\hat{G}$	95% CI
(187, 66, 193, 54)	-12.0954	(-12.107699, -12.083101)	-12.2236	(-12.236965, -12.210235)	-12.3205	(-12.334414, -12.306586)
(132, 113, 251, 0)	-12.0824	(-12.094346, -12.070454)	-12.2121	(-12.225005, -12.199195)	-12.3153	(-12.328765, -12.301835)
(95, 21, 251, 133)	-12.0951	(-12.107628, -12.082572)	-12.2304	(-12.243916, -12.216884)	-12.3107	(-12.325012, -12.296388)
(89, 73, 222, 116)	-12.0949	(-12.107252, -12.082548)	-12.2232	(-12.236734, -12.209666)	-12.3252	(-12.339347, -12.311053)
(111, 88, 176, 116)	-12.09	(-12.102403, -12.077597)	-12.2421	(-12.255275, -12.228925)	-12.3231	(-12.337081, -12.309119)
(0, 0, 203, 297)	-12.0757	(-12.087633, -12.063767)	-12.1973	(-12.210506, -12.184094)	-12.2996	(-12.313371, -12.285829)
(0, 0 , 272, 226)	-12.0768	(-12.089016, -12.064584)	-12.2112	(-12.224528, -12.197872)	-12.3089	(-12.322774, -12.295026)
(53, 42, 280, 125)	-12.0892	(-12.101602, -12.076798)	-12.2238	(-12.237327, -12.210273)	-12.3221	(-12.336196, -12.308004)
(158, 0, 174, 168)	-12.1008	(-12.112993, -12.088607)	-12.2181	(-12.231578, -12.204622)	-12.3201	(-12.334091, -12.306109)

This case study brings up several interesting and important issues. First, there are some systems that are extremely reliable, specifically those with many redundancies. The candidate evaluation methodology from Chapter 2 needed to be modified to accommodate such a system. In Chapter 2, and all of the other examples and case studies discussed in this work, the entropy of the system reliability posterior distribution given the first-stage data and the expected entropy of all candidate allocations were calculated by constructing a histogram over the entire possible reliability range (i.e., the interval (0, 1)). However, the system reliability posterior distribution draws for the LPCI system only cover a small range of that interval (i.e., roughly (0.9997945, 1)). In such instances, constructing a histogram over the entire interval (0, 1) is inefficient, as the entire posterior distribution of system reliability given the first-stage data will be lumped into a single bin near 1. This was overcome by constructing the histogram over a range slightly wider than the range of the LPCI system reliability posterior distribution (given the first-stage data) draws. Even after this adjustment was made, there was still some concern about the amount of numerical error present in the entropy calculation, and thus three different bin widths were used to compute the estimate of expected entropy for a candidate allocation.

This case study also highlights the importance of performing an optimization to find an informative candidate allocation, rather relying on intuition alone. Based on the first-stage data, and the similarity in testing cost for the basic components, it was suspected that a candidate allocation using resources components with first-stage failures was the "obvious" choice to be the most informative allocation. However, the results from multiple runs of the genetic algorithm, with different bin widths, suggest that this was not the case. The genetic algorithm identified a group of "system-critical" components, and suggested that using resources on tests of these components would provide more information about the reliability of the system than would the more "obvious" allocation.

Lastly, the average computation time for these nine GA runs (i.e., three runs for each bin width) was 1.38 days, with standard deviation 0.058 days. Once again this illustrates the computational efficiency with which our approach can be used with a genetic algorithm to

find a good resource allocation for a second-stage experiment. Thus, our approach can be considered a viable alternative to the current approach.

CHAPTER 5. SUMMARY AND DISCUSSION

Resource allocation for system reliability studies uses available first-stage data and information to choose the second-stage experiment that is expected to yield the most new information about the reliability of the system. The LPCI case study discussed in Section 4.2 illustrated that the intuitive choice for the second-stage experiment is not necessarily the most informative. The approach used by Hamada *et al* (2004), Wilson *et al* (2006), and Hamada *et al* (2008) for finding an optimal resource allocation is time-consuming and computationally intensive. In this chapter we summarize the methodology presented in this work and discuss how it makes the problem of finding an optimal, or nearly optimal, resource allocation for complex system reliability studies more manageable. Limitations of the new methodology and ideas for future investigation and related work are also noted.

5.1 New Methodology for Candidate Evaluation (Chapter 2)

In Chapter 2 we introduce computationally efficient methodology for evaluating candidate resource allocations. By evaluating candidate allocations we mean determining the expected amount of information to be gained about system reliability by performing the candidate experiment. We use posterior entropy as our planning criterion, as finding an allocation that minimizes the expected posterior entropy is equivalent to finding an allocation that maximizes the expected gain in Shannon information under the Bayesian experimental design framework.

The crux of the methodology is the realization that a specific outcome $\mathbf{x}_2$ from a candidate allocation $\mathbf{n}_2$ is comprised of independent pass/fail tests on the various system components and, for that outcome, the updated system reliability posterior distribution can be written as

$$f(\theta|\mathbf{x}_1, \mathbf{x}_2) = \frac{f(\theta|\mathbf{x}_1)p(\mathbf{x}_2|\theta)}{p(\mathbf{x}_2|\mathbf{x}_1)}$$

where the three factors in the updated system reliability posterior distribution can be estimated using the results of a single MCMC analysis based on the first-stage data. The entropy of the updated system reliability posterior distribution can then be computed as

$$\tilde{H}(\theta|\mathbf{x}_1, \mathbf{x}_2) = -\sum_{\theta} h\tilde{f}(\theta|\mathbf{x}_1, \mathbf{x}_2)\log\left(\tilde{f}(\theta|\mathbf{x}_1, \mathbf{x}_2)\right),$$

where the summation is over a regular grid of θ values.

This can be used in one of two ways. If, for a given candidate allocation $\mathbf{n}_2$, the number of possible outcomes is not too large and can be enumerated, we can compute the expected entropy for allocation $\mathbf{n}_2$ as

$$\begin{aligned}\tilde{G}(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2) &= E_{\mathbf{x}_2|\mathbf{x}_1}\tilde{H}(\theta|\mathbf{x}_1, \mathbf{x}_2) \\ &= \sum_{\mathbf{x}_2}\tilde{p}(\mathbf{x}_2|\mathbf{x})\tilde{H}(\theta|\mathbf{x}_1, \mathbf{x}_2).\end{aligned}$$

If it is infeasible to enumerate all outcomes for a candidate allocation, we can estimate the expected entropy by drawing outcomes $\mathbf{x}_2^{(1)}$, $\mathbf{x}_2^{(2)}$, ..., $\mathbf{x}_2^{(N)}$ from the distribution of $\mathbf{x}_2$ given the first-stage data and computing

$$\hat{G}(\theta|\mathbf{n}_1, \mathbf{x}_1, \mathbf{n}_2) = \frac{1}{N}\sum_{j=1}^{N}\tilde{H}(\theta|\mathbf{x}_1, \mathbf{x}_2^{(j)}).$$

Using this methodology, multiple candidates can be evaluated efficiently with the posterior draws from a single MCMC analysis of the first-stage data.

In this work, we consider a simplification of the resource allocation problem. Throughout we make the assumption that the system and its components do not age over time. In reality, the age of the components and other covariates that are related to the reliability of the system may be available. It would be extremely useful to extend the methodology from Chapter 2 to apply in such situations.

In Chapter 2, we (approximately) assess the amount of numerical error present in the histogram-based calculation of the expected entropy. It seems, in studies of this nature, consistent ranking of the candidate allocations is more important than the actual computed value for entropy. That it, it is more important to correctly identify a good allocation than it is to

compute the expected entropy for that allocation very precisely. Thus, in future work a more rigorous investigation of numerical error in the difference between expected entropy between candidate allocations would be useful.

The LPCI system case study considered in Section 4.2 highlighted an aspect of this methodology that could be improved. Throughout this work, candidates are evaluated by constructing a histogram over the entire interval (0, 1). For many of the examples considered, this was not problematic; the LPCI case study is the exception. The LPCI system is a safety feature in certain nuclear-powered boiling-water reactors (Martz and Waller (1990)); this system was found to be an extremely reliable system. Using the model of Johnson *et al* (2003) described in Section 1.2, we found a 95% credible interval for the system's demand availability to be (0.999976, 0.999999998). A histogram constructed over the entire interval (0, 1) places all system reliability posterior draws in only a few bins, leading to very erroneous entropy calculations. To rectify this situation, we constructed the histogram over an interval that was slightly wider than range of the system reliability posterior draws. Our methodology could be potentially improved by dynamically chosing the interval over which the histogram should be constructed by considering the range of the posterior draws, rather than automatically constructing the bins over the entire interval (0, 1); using the interval (0, 1) is straight-forward and worked well in most applications, but there are situations (e.g., the LPCI system) for which it fails.

5.2 Genetic Algorithms for Resource Allocation (Chapter 3)

In Chapter 3 we introduced genetic algorithms (GAs) and their application to the resource allocation problem. GAs create candidate allocations that need to be evaluated at each generation. We find that the methodology from Chapter 2 fits naturally into this framework. We consider both small problems, in which all candidates can evaluated by enumerating all possible outcomes for that candidate allocation, and large problems, where the sampling-based candidate evaluation needs to be used. The sampling-based GAs tended to be faster, even though candidates were re-evaluated each time that they generated. With both versions of the GA, prudent implementation involves multiple runs of the GA to find good allocations.

The current implementation of the genetic algorithm can easily be extended by allowing the ability to incorporate additional constraints and thus be applicable to a wider range of problems. For example, the GA can be extended to incorporate a constraint that requires a certain percentage of the budget be used to collect tests of a specific component. Additionally, the overall search performed by the GA could potentially be improved by combining the current random generation of the initial population with additional candidates that are well spread out across the solution space. Inclusion of such non-random starting solutions should broaden the scope the search and could potentially help the GA avoid local optima.

In our implementation of the sampling-based GA, we evaluated a candidate each time it was generated and treated each occurrence as a "unique" candidate and expected entropy; that is, a candidate may appear multiple times in a generation, each occurrence with a different estimate of the expected entropy. As a result, only the most informative (i.e., most negative) estimates for a candidate survive to subsequent generations of the GA. Thus, the candidates in the final population tend to look slightly more informative than they actually are (see, for example, Table 3.3) and a candidate that appears multiple times in the final population will have multiple estimates of the expected entropy. Another option that may be worth investigating is how to "update" the estimated expected entropy for a candidate allocation when it is evaluated multiple times and then report the "updated" estimate for each occurrence of the candidate allocation. An approach like this may prevent candidate allocations from looking more informative than they are and potentially lead to clear separation between candidates in the final population of the GA.

In conclusion, the methodology presented in this work can be used to make finding an optimal allocation in resource allocation studies more manageable. The methodology allows candidate allocations to be evaluated quickly and efficiently, and we have demonstrated how naturally the methodology fits with the optimization strategy currently employed (i.e., genetic algorithms). We have indicated several ways in which this methodology could be expanded or improved to make it applicable to a wider range of problems in which it is desired to find a good candidate for performing a second-stage experiment.

APPENDIX A. OUTLINE OF GENETIC ALGORITHM FOR RESOURCE ALLOCATION

Here we present an outline of the implementation of the genetic algorithm. In this outline we assume that (a) there are $k+1$ components in the system (including the system itself), (b) the candidate allocations are represented as an integer-valued vector of length $k+1$, (c) the budget is B, and (d) the component tests costs are c_i, $i = 1, \ldots, k+1$.

0. Generate initial population (i.e., "Generation $g = 0$") of m solutions. To generate a single random solution:

   (a) Generate a vector $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_{k+1})$ from the $(k+1)$-dimensional Dirichlet distribution with parameter vector α=($\frac{1}{k+1}$,$\frac{1}{k+1}$,...,$\frac{1}{k+1}$). In this vector, Z_i represents the proportion of the budget allocated to tests on Component i.

   As noted in Section 3.2, the vector $\mathbf{Z}$ is generated by making $k+1$ independent draws (Y_1, Y_2, ..., Y_{k+1}) from Gamma($\frac{1}{k+1}$,1) and computing $Z_i = \frac{Y_i}{\sum Y_i}$.

   (b) Compute the dollar amount allocated to tests on Component i to be $D_i = B \times Z_i$.

   (c) The number of tests to be performed on Component i is $n_{2,i} = \text{floor}(D_i/c_i)$.

   (d) The randomly selected candidate allocation is $\mathbf{n}_2 = (n_{2,1}, n_{2,2}, \ldots, n_{2,k}, n_{2,k+1} = n_{2,\theta})$.

1. Determine the fitness of the m candidates in Generation $g = 0$ by computing the expected entropy. If the candidates that the GA will generate are small, the expected entropy can be computed as described in Section 2.3; if the candidates will tend to be large, the sampling-based evaluation method from Section 2.6 should be used. Only one evaluation procedure (i.e., full enumeration or sampling) should be used in a single GA.

2. Rank the candidates in Generation $g = 0$ according to their expected entropy (if full enumeration is used) or estimate of expected entropy (if sampling is used); the top-ranked candidate allocation should have the lowest (most negative) entropy.

3. Use the candidates in Generation g to create m children via uniform crossover. To create one child allocation:

   (a) Choose two candidates from Generation g at random to be parent allocation. The probability that a candidate is selected to be a parent is inversely proportional to its rank.

   Denote the first parent allocation as $(n_{2,1}^{(1)}, n_{2,2}^{(1)}, \ldots, n_{2,k+1}^{(1)})$ and the second parent as $(n_{2,1}^{(2)}, n_{2,2}^{(2)}, \ldots, n_{2,k+1}^{(2)})$.

   (b) For each entry of the child allocation $n_{2,l}$, $l = 1, 2, \ldots, k+1$, draw b from Binomial(1,0.5) and set

$$n_{2,l} = \begin{cases} n_{2,l}^{(1)} & \text{if } b = 0 \\ n_{2,l}^{(2)} & \text{if } b = 1 \end{cases}$$

   If the child allocation exceeds the budget, it is repaired by selecting a component with a nonzero sample size at random and decreasing it's sample size by one; this is repeated until the child allocation satisfies the budget.

4. Use candidates in Generation g to create m mutation allocations, one for each candidate allocation in Generation g; this is done using the approach described by Hamada *et al* (2004). To create the mutation allocation for Candidate j, modify each entry of $\mathbf{n}_2^{(j)} = (n_{2,1}^{(j)}, n_{2,2}^{(j)}, \ldots, n_{2,k+1}^{(j)})$; to modify $n_{2,i}^{(j)}$,

   (a) Compute $z_i = \frac{n_{2,i}^{(j)} - L_i}{U_i - L_i}$, where L_i in the minimum number of tests allowed for Component i and U_i is the maximum number of tests allowed (i.e., $U_i = \text{floor}(B/c_i)$).

   (b) Compute $d_i = \log[z_i/(1-z_i)] + [Uniform(0,1) - 0.5]\sigma\exp(-\mu g)$, where σ controls the rate at which the variation decreases with generation (Hamada *et al* (2004)); we use $\sigma = 1.25$.

(c) Compute the mutated sample size as

$$n_{2,i}^{(j)'} = \text{floor}\left(L_i + (U_i + 1 - L_i)\frac{\exp(d_i)}{1+\exp(d_i)}\right).$$

The mutation allocation for Candidate j is $(n_{2,1}^{(j)'}, n_{2,2}^{(j)'}, \ldots, n_{2,k+1}^{(j)'})$. If the mutation solution exceeds the budget, it is repaired by selecting a component with a nonzero sample size at random and decreasing it's sample size by one; this is repeated until the mutation allocation satisfies the budget.

5. Evaluate the m recombination solutions and m mutation solutions by computing the expected entropy (for small problems) or an estimate of the expected entropy (for large problems).

6. Sort the $3m$ allocations according to their fitness. Keep the m most informative allocations (i.e., those with the lowest expected entropy) to be the population in Generation $g+1$.

7. Proceed to Generation $g+1$ and repeat Steps 3-6 for specified number of generations, G.

APPENDIX B. ADDITIONAL MATERIAL FOR MISSILE CASE STUDY

In Section 4.1 we consider finding an optimal allocation of resources for the series missile system described by Martz *et al* (1988). Here we provide additional material for the missile case study, including the system details as described by Martz *et al* (1988). The initial analysis of these data by Martz *et al* (1988) is described briefly. We then discuss how the data can be analyzed using the fully Bayesian approach of Johnson *et al* (2003) and include details of the MCMC implementation, diagnostics, and model assessment.

System Details

The air-to-air heat-seeking missile system consists of five major subsystems connected in series. The major subsystems and the basic components that comprise them are displayed in Table B.1. The components that comprise the warhead are generically labeled A-I to prevent security classification problems (Martz *et al* (1988)). The number in parentheses indicates the component number, as assigned in the event tree in Figure 4.1. We note that the first three subsystems detailed in Table B.1 and the basic components comprising these systems are the only components identified as being testable; in the resource allocation phase, candidate allocations can only consist of tests of these subsystems and their basic components.

The available current binomial test data and expert best guesses are displayed in Table 4.1. The number of tests performed ranges from 5 to 330. The basic components in the aircraft subsystem and the aircraft subsystem itself were tested the most, with between 130 and 300 tests per component. The fewest current binomial tests were performed on the warhead subsystem and the basic components in the warhead subsystem, with most components having 40

or fewer tests. The expert best guesses provided by Martz *et al* (1988) can be interpreted as the number of historical successes and historical tests. Very little historical data are available for the warhead subsystem and its basic components, while the system, the C^3I subsystem, and the components in the C^3I and maintenance/logistics subsystems only have historical data available.

Table B.1 Major subsystems and basic components in air-to-air heat-seeking missile system (Table 1 from Martz *et al* (1988))

Warhead (33)	Missile (34)	Aircraft (35)	C^3I^a (36)	Logistics/ Maintenance (37)
A (1)	Power Supply (10)	Flight structure (16)	Airspace control effectiveness (25)	Ground handling (30)
B (2)	Target acquisition/ guidance system (11)	Avionics (17)	Rules of engagement (26)	Storage (31)
C (3)	Motor (12)	Power (18)	Identification friend or foe/ visual (27)	Missile availability (32)
D (4)	Flight Structure (13)	Flight Control (19)	Aircraft on-station availability (28)	
E (5)	Aircraft Interface (14)	Environmental (20)	Radio communications (29)	
F (6)	Control (15)	Acquisition/ Fire Control (21)		
G (7)		Launching (22)		
H (8)		Missile Interface (23)		
I (9)		Human Intervention (24)		

[a] C^3I is a military acronym that stands for "Command, Control, Communications, and Intelligence"

Analysis of Martz *et al* (1988)

In the analysis of Martz *et al* (1988), the expert prior information stems from historical data available at various levels of the system. At the basic component level, if historical data

are available, the prior distribution for the reliability of the j^{th} component in the i^{th} subsystem is Beta($s^0_{ij}+1$, $n^0_{ij}-s^0_{ij}+1$), where, in the case that s^0_{ij} and n^0_{ij} are non-negative integers, s^0_{ij} is interpreted as one less than the number of historical component successes and n^0_{ij} is interpreted as two less than the number of historical component tests. If historical data are not available for the j^{th} component in the i^{th} subsystem, a non-informative prior distribution should be used for the reliability of this basic component; Martz *et al* (1988) note that $n^0_{ij}=s^0_{ij}$ would be interpreted as one success in two historical component tests and would yield the Beta(1, 1) prior, while $n^0_{ij}=-1$ and $s^0_{ij}=-\frac{1}{2}$ yields the Jeffrey's prior (Beta($\frac{1}{2}$, $\frac{1}{2}$)), but the historical success/prior test interpretation does not hold. At the subsystem level, if historical data are available for subsystem i, the prior distribution for the reliability of Subsystem i is Beta(s^0_i+1, $n^0_i-s^0_i+1$), where s^0_i and n^0_i are required to be non-negative integers and can be interpreted as above. Similarly, if historical data are available at the system level, the prior distribution for the system reliability is Beta(s^0+1, n^0-s^0+1) with s^0 and n^0 restricted to being non-negative integers. These component reliability prior distributions are the *native* prior distributions. If no historical data are available for a non-basic component, no native prior is specified for the reliability of that component.

The analysis of Martz *et al* (1988) is a two-stage analysis, the first stage completed at the subsystem-level and the second stage at the system-level. We first summarize the subsystem-level analysis; suppose Subsystem i is comprised of k_i basic components in series. The first step is to compute the basic component reliability posterior distribution for each of the k_i basic components in Subsystem i. Combining the prior distribution for the reliability of the j^{th} component in the i^{th} subsystem and the current binomial data for that basic component, yields the posterior distribution Beta($s_{ij}+s^0_{ij}+1$, $n_{ij}+n^0_{ij}-s_{ij}-s^0_{ij}+1$), where s_{ij} is the number of successes out of n_{ij} trials for the j^{th} component in the i^{th} subsystem. If there are no current binomial data available for this basic component, the posterior distribution reduces to the prior distribution for the reliability of this basic component. If no historical data are available, a non-informative prior distribution, such as Beta(1, 1) or Beta($\frac{1}{2}$, $\frac{1}{2}$), is used.

The next step in the subsystem-level analysis is to compute the *induced* prior distribution

Table B.2 Induced, Native, and Combined Prior Distributions of Martz *et al* (1988) and Posterior Distributions for Major Subsystems and Missile System

Component	Prior Distribution			Posterior
	Induced	Native	Combined	Distribution
Warhead (33)	Beta(10.64, 3.51)	—	Beta(10.64, 3.51)	Beta(18.64, 3.51)
Missile (34)	Beta(405.11, 67.68)	—	Beta(405.11, 67.68)	Beta(412.11, 68.68)
Aircraft (35)	Beta(419.46, 43.03)	Beta(258, 13)	Beta(338.73, 28.01)	Beta(529.73, 42.01)
C^3I (36)	Beta(500.88, 40.75)	Beta(56, 12)	Beta(167.22, 19.19)	Beta(167.22, 19.19)
Logistics/ Mainten. (37)	Beta(1021.76, 109.03)	—	Beta(1021.76, 109.03)	Beta(1021.76, 109.03)
System (38)	Beta(48.52, 41.06)	Beta(116, 151)	Beta(82.26, 96.03)	Beta(82.26, 96.03)

for the reliability of Subsystem i (Martz *et al* (1988)). Because the k_i basic components in Subsystem i are connected in series to form Subsystem i and thus the reliability of Subsystem i is the product of the basic component reliabilities, the induced prior distribution for the reliability of Subsystem i is the product of k_i independent Beta random variables, where the j^{th} random variable has the Beta posterior distribution described above. Martz *et al* (1988) approximate the induced Subsystem i reliability prior distribution with a Beta distribution with parameters a_i and b_i using the approach of Springer (1979) and Thompson and Haynes (1980).

Next, Martz *et al* (1988) average the induced and native Subsystem i reliability prior distributions to obtain the *combined* Beta prior distribution for Subsystem i's reliability. This is done using the method of Winkler, R. L. (1968) in which the native prior distribution is treated as being proportional to a binomial likelihood. The native and induced prior distributions are then weighted to obtain the combined prior distribution for Subsystem i's reliability, Beta($w_{i1}a_i+w_{i2}s_i^0+w_{12}$, $w_{i1}b_i+w_{i2}n_i^0-w_{i2}s_i^0+w_{i2}$) where w_{i1} and w_{i2} are weights selected such that they sum to 1 (Martz *et al* (1988)). The combined Subsystem i prior distribution is updated with any current binomial data available for Subsystem i to obtain the posterior distribution of Subsystem i's reliability, Beta($w_{i1}a_i+w_{i2}s_i^0+s_i+w_{12}$, $w_{i1}b_i+w_{i2}n_i^0-w_{i2}s_i^0+n_i-s_i+w_{i2}$). This is repeated for each subsystem.

The system-level analysis proceeds similarly, by using the subsystems' posterior distributions to construct an induced prior distribution for the system reliability. This induced prior distribution is averaged with the native prior distribution to yield the combined system reliabil-

ity prior distribution. The combined system reliability prior distribution and any system-level current binomial data are used to obtain the system reliability posterior distribution. The induced, native and combined prior distributions and posterior distributions for the reliability of the five subsystems and the missile system are displayed Table B.2. Due to the absence of system-level data, the system reliability posterior distribution is the system reliability prior distribution. The system reliability posterior mean is approximately 0.46; the standard deviation of the posterior distribution is 0.037 and the posterior entropy is -1.871771.

Analysis of First-Stage Data Using Model of Johnson *et al* (2003)

We analyze the missile system data using the model described in Section 1.2. Under the model of Johnson *et al* (2003), the joint posterior distribution is given by

$$\begin{aligned} g(\mathbf{p}, N, \gamma, J|\mathbf{x}, \mathbf{n}, \pi, \alpha, \beta, \psi, \omega, \tau, \phi) \quad &\propto \quad \prod_{i \in S_0} p_i^{x_i}(1-p_i)^{n_i - x_i} \\ &\times \quad \prod_{i \in S_1} B(p_i; N\pi_i + 1, N(1-\pi_i) + 1) \\ &\times \quad \prod_{i=1}^{32} B(p_i; J\gamma, J(1-\gamma)) \\ &\times \quad G(N; \alpha, \beta) B(\gamma; \psi, \omega) G(J; \tau, \phi). \end{aligned}$$

The prior distributions for N and J were specified to be Gamma(5, 1); note these prior distributions were used by Johnson *et al* (2003) and Anderson-Cook *et al* (2007) and have the interpretation that, on average, the expert's best guess is worth 5 system tests. The non-informative Jeffrey's prior distribution, Beta($\frac{1}{2}$, $\frac{1}{2}$), was specified for γ.

A component-wise random-walk Metropolis-Hastings algorithm was used to explore the joint posterior distribution; that is, the parameter vector ($\mathbf{p}$, , N, , J , γ) is updated in four "blocks". The logistic regression of Graves (2005) was used to specify the standard deviation of the proposal distributions for each of the four blocks such that the acceptance rate for each block would be roughly 0.25. The standard deviations for the four blocks were specified to be 0.24, 1.34, 0.9, and 1.15. The steps of the Metropolis-Hastings algorithm, which is implemented in C, are outlined below.

0. Set $j = 0$ and draw initial values for $\mathbf{p}$, N, γ and J

   a. For each basic component, draw $z_i^{(0)}$ from N(0, σ_z=0.24), $i = 1, \ldots, 32$ and compute the basic component reliabilities

   $$p_i^{(0)} = \frac{\exp(z_i^{(0)})}{1 + \exp(z_i^{(0)})}, \qquad i = 1, \ldots, 32,$$

   then compute the non-basic component reliabilities using the appropriate basic component reliabilities.

   b. Draw $u^{(0)}$ from N(0, σ_u=1.34) and compute $N^{(0)} = \exp(u^{(0)})$.

   c. Draw $v^{(0)}$ from N(0, σ_v=0.9) and compute $\gamma^{(0)} = \exp(v^{(0)})/(1 + \exp(v^{(0)}))$.

   d. Draw $w^{(0)}$ from N(0, σ_w=1.15) and compute $J^{(0)} = \exp(w^{(0)})$.

1. Generate proposals for the component reliabilities ($\mathbf{p}^*$) by drawing z_i^* from N($z_i^{(j-1)}$, σ_z), $i = 1, \ldots, 32$ and computing the proposals for the basic component reliabilities as

   $$p_i^* = \frac{\exp(z_i^*)}{1 + \exp(z_i^*)}, \qquad i = 1, \ldots, 32$$

   and finding the proposals for the non-basic component reliabilities using the appropriate basic component reliabilities.

   Note that

   $$f(p_i^*|p_i^{(j-1)}) = \frac{1}{p_i^*(1-p_i^*)} \frac{1}{\sqrt{2\pi}} \frac{1}{\sigma_z} \exp\left(-\frac{1}{2\sigma_z^2} \left(\log(\frac{p_i^*}{1-p_i^*}) - \log(\frac{p_i^{(j-1)}}{1-p_i^{(j-1)}}) \right)^2 \right)$$

   and the proposals for the basic component reliabilities are drawn independently.

2. Compute

   $$\begin{aligned} r &= \frac{g(\mathbf{p}^*, N^{(j-1)}, \gamma^{(j-1)}, J^{(j-1)}|\mathbf{x}, \mathbf{n}, \boldsymbol{\pi}, \alpha, \beta, \psi, \omega, \tau, \phi)}{g(\mathbf{p}^{(j-1)}, N^{(j-1)}, \gamma^{(j-1)}, J^{(j-1)}|\mathbf{x}, \mathbf{x}, \boldsymbol{\pi}, \alpha, \beta, \psi, \omega, \tau, \phi)} \frac{f(\mathbf{p}^{(j-1)}|\mathbf{p}^*)}{f(\mathbf{p}^*|\mathbf{p}^{(j-1)})} \\ &= \frac{g(\mathbf{p}^*, N^{(j-1)}, \gamma^{(j-1)}, J^{(j-1)}|\mathbf{x}, \mathbf{n}, \boldsymbol{\pi}, \alpha, \beta, \psi, \omega, \tau, \phi)}{g(\mathbf{p}^{(j-1)}, N^{(j-1)}, \gamma^{(j-1)}, J^{(j-1)}|\mathbf{x}, \mathbf{n}, \boldsymbol{\pi}, \alpha, \beta, \psi, \omega, \tau, \phi)} \frac{\prod_{i=1}^{32} p_i^*(1-p_i^*)}{\prod_{i=1}^{32} p_i^{(j-1)}(1-p_i^{(j-1)})}. \end{aligned}$$

3. Draw ν from Uniform(0, 1). If $\nu \leq r$, set $\mathbf{p}^{(j)} = \mathbf{p}^*$; otherwise, $\mathbf{p}^{(j)} = \mathbf{p}^{(j-1)}$.

4. Generate a proposal N^* by drawing u^* from N($u^{(j-1)}$, σ_u) and compute $N^* = \exp(u^*)$. Note that

$$f(N^*|N^{(j-1)}) = \frac{1}{N^*}\frac{1}{\sqrt{2\pi}}\frac{1}{\sigma_u}\exp\left(\frac{1}{2\sigma_u^2}\left(\log(N^*) - \log(N^{(j-1)})\right)^2\right).$$

5. Compute

$$r = \frac{g(\mathbf{p}^{(j)}, N^*, \gamma^{(j-1)}, J^{(j-1)}|\mathbf{x}, \mathbf{n}, \boldsymbol{\pi}, \alpha, \beta, \psi, \omega, \tau, \phi)}{g(\mathbf{p}^{(j)}, N^{(j-1)}, \gamma^{(j-1)}, J^{(j-1)}|\mathbf{x}, \mathbf{n}, \boldsymbol{\pi}, \alpha, \beta, \psi, \omega, \tau, \phi)}\frac{N^*}{N^{(j-1)}}.$$

6. Draw ν from Uniform(0, 1). If $\nu \leq r$, set $N^{(j)} = N^*$; otherwise, set $N^{(j)} = N^{(j-1)}$.

7. Generate a proposal γ^* by drawing v^* from N($v^{(j-1)}$, σ_v) and compute

$$\gamma^* = \exp(v^*)/(1 + \exp(v^*)).$$

8. Compute

$$\frac{g(\mathbf{p}^{(j)}, N^{(j)}, \gamma^*, J^{(j-1)}|\mathbf{x}, \mathbf{n}, \boldsymbol{\pi}, \alpha, \beta, \psi, \omega, \tau, \phi)}{g(\mathbf{p}^{(j)}, N^{(j)}, \gamma^{(j-1)}, J^{(j-1)}|\mathbf{x}, \mathbf{n}, \boldsymbol{\pi}, \alpha, \beta, \psi, \omega, \tau, \phi)}\frac{\gamma^*(1-\gamma^*)}{\gamma^{(j-1)}(1-\gamma^{(j-1)})}.$$

9. Draw ν from Uniform(0, 1). If $\nu \leq r$, set $\gamma^{(j)} = \gamma^*$; otherwise, set $\gamma^{(j)} = \gamma^{(j-1)}$.

10. Generate a proposal J^* by drawing w^* from N($w^{(j-1)}$, σ_w) and computing $J^* = \exp(w^*)$.

11. Compute

$$\frac{g(\mathbf{p}^{(j)}, N^{(j)}, \gamma^{(j)}, J^*|\mathbf{x}, \mathbf{n}, \boldsymbol{\pi}, \alpha, \beta, \psi, \omega, \tau, \phi)}{g(\mathbf{p}^{(j)}, N^{(j)}, \gamma^{(j)}, J^{(j-1)}|\mathbf{x}, \mathbf{n}, \boldsymbol{\pi}, \alpha, \beta, \psi, \omega, \tau, \phi)}\frac{J^*}{J^{(j-1)}}.$$

12. Draw ν from Uniform(0, 1). If $\nu \leq r$, set $J^{(j)} = J^*$; otherwise, set $J^{(j)} = J^{(j-1)}$.

13. Increment j and return to Step 1. Repeat for a specified number of iterations.

The MCMC algorithm was run for 10,025,000 iterations. Trace plots (not included) indicated that a burn-in period of 25,000 iterations was sufficient and thus the first 25,000 draws were discarded. Figures B.1-B.3 display autocorrelation plots for the 41 model parameters (38 component reliabilities, N, γ, and J); many parameters show substantial autocorrelation even with a lag of 500, thus the draws were thinned to every 1000^{th} draw. A total of 10,000 posterior draws were collected.

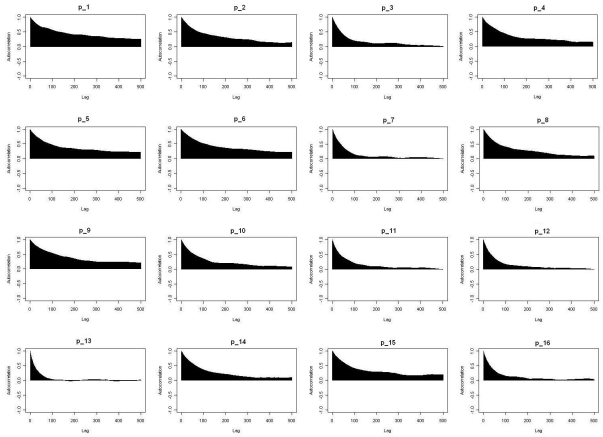


Figure B.1 Autocorrelation plots for Basic Components 1-16

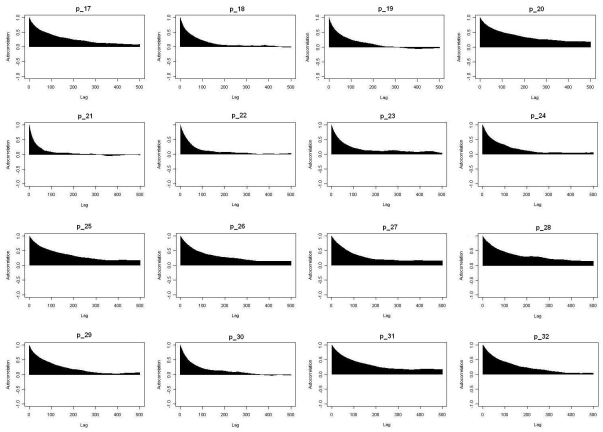


Figure B.2 Autocorrelation plots for Basic Components 17-32

Density estimates of the marginal posterior distributions for 41 model parameters are displayed in Figures B.4-B.6. The posterior distributions for the model parameters are further summarized in Table B.3. The estimated posterior means, standard deviations, and posterior quantiles are reported with naive and batch means Monte Carlo standard error estimates (Jones *et al* (2006)). Both standard error estimates were computed using the coda package in R (Plummer *et al* (2008)).

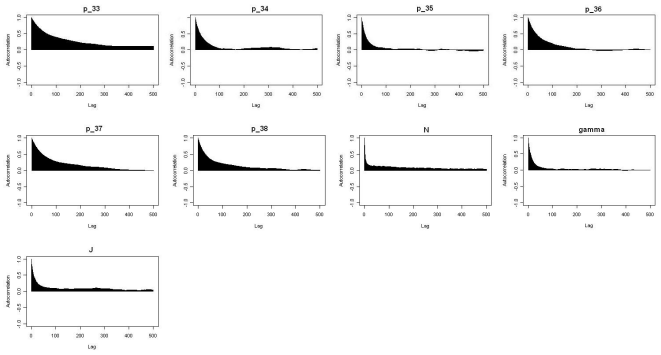


Figure B.3 Autocorrelation plots for Subsystems, N, γ, and J

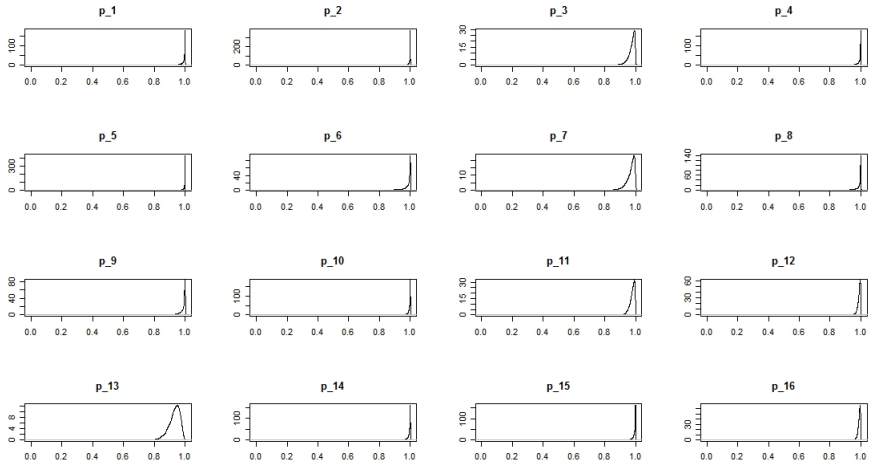


Figure B.4 Density estimates for Basic Components 1 - 16

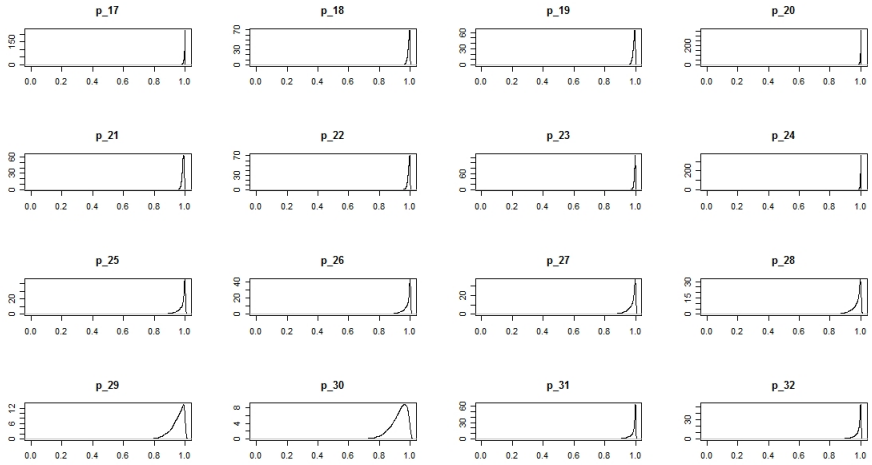


Figure B.5 Density estimates for Basic Components 17 - 32

The system reliability (p_{38}) posterior mean is estimated to be 0.55, higher than that of Martz *et al* (1988). The estimated standard deviation of the system reliability posterior distribution is 0.063, which is nearly twice that of Martz *et al* (1988). Martz *et al* (1988) note that the narrow posterior distribution for the missile system reliability is likely due to the strong historical data that were used in their analysis.

The posterior mean of the precision parameter N is estimated to be 15.1, indicating that, on average, the expert's judgment is worth roughly 15 system tests. Johnson *et al* (2003)

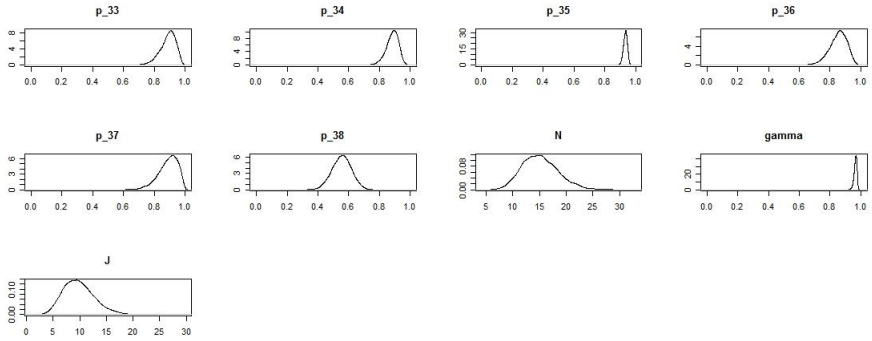


Figure B.6 Density estimates for Subsystems, System, N, γ, and J

and Anderson-Cook *et al* (2007) note that as the posterior mean of N is larger than the prior mean (5), the expert appears to be rather knowledgeable about the system.

Table B.3 Estimated posterior means, standard deviations, and quantiles, with naive and batch means Monte Carlo SE estimates from the R package coda, for the 41 model parameters

Parameter	Mean	Std. Dev.	Naive SE	Batch Means SE	Quantiles		
					2.5%	50%	97.5%
p_1	0.9926	0.012681	1.27E-04	1.26E-04	0.9565	0.9979	1
p_2	0.9964	0.006237	6.24E-05	6.86E-05	0.9781	0.999	1
p_3	0.975	0.021309	2.13E-04	2.16E-04	0.9188	0.9809	0.9985
p_4	0.9925	0.01272	1.27E-04	1.31E-04	0.9548	0.998	1
p_5	0.9969	0.00497	4.97E-05	4.79E-05	0.9822	0.9991	1
p_6	0.9859	0.023859	2.39E-04	2.70E-04	0.9112	0.9962	1
p_7	0.9698	0.025434	2.54E-04	2.67E-04	0.9047	0.9767	0.9984
p_8	0.9905	0.01634	1.63E-04	1.75E-04	0.9426	0.9973	1
p_9	0.9831	0.029102	2.91E-04	3.14E-04	0.8958	0.9955	1
p_{10}	0.9937	0.009579	9.58E-05	9.57E-05	0.9656	0.9975	1
p_{11}	0.9783	0.017071	1.71E-04	2.08E-04	0.9355	0.9825	0.9986
p_{12}	0.9886	0.009655	9.66E-05	8.97E-05	0.9634	0.9912	0.9993
p_{13}	0.931	0.035712	3.57E-04	4.03E-04	0.8459	0.9367	0.9836
p_{14}	0.9932	0.009736	9.74E-05	8.92E-05	0.9652	0.997	1
p_{15}	0.9935	0.008934	8.93E-05	9.90E-05	0.9674	0.9969	1
p_{16}	0.9897	0.007623	7.62E-05	8.02E-05	0.9704	0.9914	0.9991
p_{17}	0.9959	0.00499	4.99E-05	5.66E-05	0.9822	0.9977	1
p_{18}	0.9901	0.007385	7.38E-05	7.33E-05	0.9715	0.9918	0.9992
p_{19}	0.9896	0.007692	7.69E-05	7.99E-05	0.9703	0.9914	0.9992
p_{20}	0.997	0.004193	4.19E-05	4.57E-05	0.9849	0.9986	1
p_{21}	0.9869	0.006684	6.68E-05	5.74E-05	0.971	0.9879	0.9967
p_{22}	0.9904	0.007299	7.30E-05	6.71E-05	0.9721	0.9921	0.9992
p_{23}	0.9946	0.004373	4.37E-05	4.61E-05	0.9831	0.9957	0.9996
p_{24}	0.9977	0.00261	2.61E-05	2.45E-05	0.9905	0.9985	1
p_{25}	0.9766	0.031362	3.14E-04	3.38E-04	0.8892	0.9896	1
p_{26}	0.9754	0.032924	3.29E-04	3.37E-04	0.8817	0.989	1
p_{27}	0.9733	0.034086	3.41E-04	2.98E-04	0.877	0.9867	1
p_{28}	0.9707	0.035521	3.55E-04	3.30E-04	0.8728	0.9841	1
p_{29}	0.9489	0.0443	4.43E-04	4.71E-04	0.8377	0.9598	0.9989
p_{30}	0.9242	0.055023	5.50E-04	5.58E-04	0.787	0.9363	0.9939
p_{31}	0.9813	0.028852	2.89E-04	2.91E-04	0.8995	0.9931	1
p_{32}	0.9791	0.030841	3.08E-04	3.14E-04	0.8908	0.9916	1
p_{33}	0.8882	0.050916	5.09E-04	5.37E-04	0.7715	0.8956	0.9661
p_{34}	0.8828	0.03931	3.93E-04	4.39E-04	0.7948	0.8869	0.9476
p_{35}	0.9337	0.012615	1.26E-04	1.25E-04	0.907	0.9345	0.9562
p_{36}	0.8528	0.055937	5.59E-04	5.85E-04	0.7291	0.8586	0.9455
p_{37}	0.888	0.064215	6.42E-04	6.16E-04	0.7358	0.8976	0.9808
p_{38}	0.5538	0.063041	6.30E-04	6.31E-04	0.4276	0.5552	0.674
N	15.1383	3.401019	3.40E-02	3.62E-02	9.2535	14.8945	22.3919
γ	0.9651	0.009885	9.89E-05	1.10E-04	0.942	0.9665	0.9803
J	9.8519	2.978742	2.98E-02	3.61E-02	4.8875	9.5775	16.49

Model Assessment and Sensitivity Analysis

To assess how well the model fit the data, the Bayesian χ^2 goodness of fit test (Johnson (2004)) was employed. The Bayesian χ^2 test is performed by constructing K equal probability bins, and for a single posterior draw, sorting the data into bins according to their cumulative probabilities. Let $\boldsymbol{\theta}$ denote a single draw of the parameter vector from the joint posterior distribution. The corresponding test statistic is

$$R(\boldsymbol{\theta}) = \frac{\sum_{k=1}^{K}(m_k(\boldsymbol{\theta}) - Np_k)^2}{Np_k},$$

where N is the number of observations in the data set, $p_k = \frac{1}{k}$ is the probability of being in Bin k, and $m_k(\boldsymbol{\theta})$ is the actual number of observations in Bin k (note that Np_k is the expected number of observations in Bin k). Johnson (2004) provides the rule-of-thumb that the number of bins be selected as $K = N^{0.4}$. This test statistic is asymptotically distributed as a χ^2_{K-1} random variable (Johnson (2004)).

To obtain the actual bin counts $m_k(\boldsymbol{\theta})$, the bins are defined to be $0 \equiv a_0 < a_1 < \cdots < a_{K-1} < a_K \equiv 1$. Then $m_k(\boldsymbol{\theta})$ is the number of observations with $F(x_i|\theta) \in (a_{k-1}, a_k]$. Johnson (2004) notes that in the case of discrete data, a continuity correction should be employed. One such correction uses $\tilde{F}(x_i|\boldsymbol{\theta})$, drawn from the Uniform$(F(x_i - 1|\boldsymbol{\theta}),\ F(x_i|\boldsymbol{\theta}))$ distribution, as the cumulative probability for x_i. Weaver and Hamada (2008) note that when $x_i = 0$, $\tilde{F}(x_i|\theta)$ is drawn from the Uniform$(0,\ F(x_i|\boldsymbol{\theta}))$ distribution.

This test statistic can be computed for each of the M draws from the joint posterior distribution, resulting in a collection of M test statistics. Johnson (2004) notes that one approach for assessing the goodness of fit is to report the proportion of the M test statistics that exceed a specified critical value of the χ^2_{K-1} distribution. If there is no model lack of fit, the proportion of test statistics exceeding the specified critical value should be approximately equal to the size of the test. A large deviation from this value indicates model lack of fit.

Weaver and Hamada (2008) provide R code for performing the Bayesian χ^2 goodness of fit test when there are binomial data. There are $N = 27$ observations in this data set; using $K = N^{0.4}$ implies 4 bins, with $p_k = 0.25$. About 8.7% of the test statistics exceeded the 95%

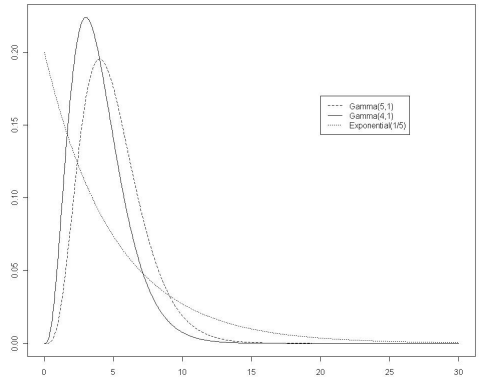


Figure B.7 Prior distributions for N and J in the sensitivity analysis

quantile of the χ^2_3 distribution, indicating that there is no problem with lack of fit.

Finally, we assess the sensitivity of the results to the choice of prior distribution for N, γ, and J. In the analysis performed here, Gamma(5, 1) priors were used for both precision parameters N and J; this same choice is made by Johnson *et al* (2003), Anderson-Cook *et al* (2007), and Hamada *et al* (2008). The parameter J is inversely proportional to the variance of the hierarchical prior distribution specified for the basic component reliabilities and N is inversely proportional to the variance of the prior distribution specified for the reliability of the components for which an expert's best guess is specified; centering the priors for J and N on smaller values implies, on average, a larger variance for the component reliability priors.

To assess the sensitivity of the system reliability posterior distribution to this choice of prior distribution for J and N we consider two other prior distributions for these parameters that also give weight to the small values: Gamma(4, 1) and Exponential(1/5), where the Exponential distribution is parameterized as

$$f(x) = \lambda e^{-\lambda x}, \qquad x > 0, \lambda > 0$$

with mean $\mu = \frac{1}{\lambda}$ and variance $\sigma^2 = \frac{1}{\lambda^2}$. The shape of the former is similar to that of the prior distribution actually used in the analysis, with a slightly lower mean and variance; the latter has the same mean as the prior distribution used in this analysis, but is more variable with a different shape. The three prior distributions are displayed in Figure B.7.

In our analysis, we use the non-informative Jeffrey's prior distribution for the parameter

Table B.4 Sensitivity of system reliability posterior distribution to choice of prior distribution for the parameters N, J, and γ

Case	N	J	γ	μ	2.5%	50%	97.5%
1	Gamma(5, 1)	Gamma(5, 1)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.5538	0.4276	0.5552	0.674
2	Gamma(5, 1)	Gamma(5, 1)	Beta(1, 1)	0.5528	0.4265	0.5534	0.6758
3	Gamma(5, 1)	Gamma(4, 1)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.5519	0.4227	0.5529	0.6740
4	Gamma(5, 1)	Gamma(4, 1)	Beta(1, 1)	0.5496	0.4219	0.5506	0.6703
5	Gamma(4, 1)	Gamma(5, 1)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.5563	0.4273	0.5574	0.6766
6	Gamma(4, 1)	Gamma(5, 1)	Beta(1, 1)	0.5528	0.4278	0.5533	0.6763
7	Gamma(4, 1)	Gamma(4, 1)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.5525	0.4218	0.5543	0.6753
8	Gamma(4, 1)	Gamma(4, 1)	Beta(1, 1)	0.5495	0.4185	0.5506	0.6722
9	Exp($\frac{1}{5}$)	Exp($\frac{1}{5}$)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.5097	0.2109	0.5206	0.7566
10	Exp($\frac{1}{5}$)	Exp($\frac{1}{5}$)	Beta(1, 1)	0.5089	0.2071	0.5213	0.7508
11	Exp($\frac{1}{5}$)	Gamma(5, 1)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.5909	0.3771	0.5964	0.7628
12	Exp($\frac{1}{5}$)	Gamma(5, 1)	Beta(1, 1)	0.5873	0.3756	0.5939	0.7596
13	Exp($\frac{1}{5}$)	Gamma(4, 1)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.5845	0.3691	0.5926	0.7607
14	Exp($\frac{1}{5}$)	Gamma(4, 1)	Beta(1, 1)	0.5785	0.3581	0.5862	0.7556
15	Gamma(5, 1)	Exp($\frac{1}{5}$)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.5366	0.3898	0.5384	0.6726
16	Gamma(5, 1)	Exp($\frac{1}{5}$)	Beta(1, 1)	0.5384	0.3926	0.5408	0.6724
17	Gamma(4, 1)	Exp($\frac{1}{5}$)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.5377	0.3882	0.5387	0.6770
18	Gamma(4, 1)	Exp($\frac{1}{5}$)	Beta(1, 1)	0.5382	0.3847	0.5404	0.6777

γ; γ represents the mode of the basic component reliability prior distribution. We consider another non-informative prior distribution for γ, Beta(1, 1). In the sensitivity analysis we consider all combinations of these priors for N, γ, and J.

Table B.4 displays the system reliability posterior mean and 2.5%, 50%, and 97.5% posterior quantiles for the 18 combinations of prior distributions for N, J, and γ; Case 1 in the table contains the results from the actual analysis. The results for Cases 2 - 8 look very similar to those of the actual analysis; these cases all involved Gamma prior distributions for N and J. When the Exponential prior is used for either J or N, or both, the system reliability posterior distribution considerably changes. In Cases 9 and 10, Exponential priors were used for both J and N; these two cases had the lowest posterior means, roughly 0.51, and the widest 95% credible sets, nearly (0.21, 0.76). In Cases 11 - 14, the Exponential prior is used for N and a Gamma prior is used for J; these four cases result in the highest system reliability posterior means. The Exponential prior was used for J in Cases 15 - 18 with a Gamma prior on N; the system reliability posterior mean was roughly 0.54, slightly lower than in the actual analysis, for these four cases.

These results suggest that using a Beta(1, 1) prior distribution, rather than the Jeffrey's

prior distribtion, for γ changes the system reliability posterior distribution very little. Similarly, using either of the Gamma distributions described here for J and N yields similar results for system reliability. The use of the Exponential prior distribution for either J or N, or both, has a much greater impact on the system reliability posterior distribution; Exponential priors on these parameters puts quite a bit of weight on small values of N and J, resulting in more variability in the basic component and expert best guess component reliability prior distributions. In all cases where an Exponential prior distribution was used, the system reliability posterior distribution was wider than when only Gamma prior distributions were used.

APPENDIX C. ADDITIONAL MATERIAL FOR LPCI SYSTEM CASE STUDY

In Section 4.2 we discuss finding an optimal resource allocation for the low-pressure coolant injection system described by Martz and Waller (1990). We briefly describe the analysis performed by Martz and Waller (1990). We also include a description of how these data can be analyzed using the model of Johnson *et al* (2003), MCMC diagnostics and model assessment.

Available Data and Information

The available data and information on the components in the LPCI system are displayed in Table 4.10. The LPCI system contains several redundant components. For example, the four pumps (A, B, C, and D) are redundant components, though each is assumed to have its own underlying availability value. The historical data for these components come from similar components in similar systems; thus the available historical data is the same for each of these four components. The same is true for the motor-operated valves (MOV-25 A, B) and the check valves (CV-48 A, B, C, D and CV-46 A, B). These 12 basic components are the only components that have current binomial data available. Historical data the for lowest subsystems in the event tree (Pump Trains, A, B, C, D and LPCI subsystems A and B) are also available. Martz and Waller (1990) note that this historical subsystem information comes from IEEE (1983), sections 11.1.2.4.2.2, 11.2.3, and 11.2.a.2 (Martz and Waller (1990)).

Table C.1 Beta posterior distributions and posterior quantiles for subsystem and LPCI system demand availability

Component	Posterior Distribution	Quantiles		
		5%	50%	95%
Pump Train A (13)	Beta(324.31, 3.55)	0.978	0.9901	0.9966
Pump Train C (14)	Beta(328.57, 2.05)	0.985	0.9948	0.9989
LPCI Subsystem A (15)	Beta(618.27, 1.91)	0.9926	0.9974	0.99949
Pump Train B (16)	Beta(350.59, 0.55)	0.9942	0.99922	0.999990
Pump Train D (17)	Beta(328.57, 2.05)	0.985	0.9948	0.9989
LPCI Subsystem B (18)	Beta(618.27, 1.91)	0.9926	0.9974	0.99949
Pump Subsystem A (19)	Beta(16625.22, 1.12)	0.99981	0.999951	0.9999955
Pump Subsystem B (20)	Beta(32444.46, 0.32)	0.999956	0.9999974	.9999999980
LPCI Train A (21)	Beta(631.26, 1.99)	0.9925	0.9974	0.99944
LPCI Train B (22)	Beta(620.20, 1.92)	0.9926	0.9974	0.99948
LPCI System (23)	Beta(80745.70, 0.78)	0.999968	0.9999940	0.99999975

Analysis of Martz and Waller (1990)

The analysis performed by Martz and Waller (1990) is a two-stage analysis similar to that described in Appendix B. Martz and Waller (1990) no longer require that the "historical data" for the subsystems and system (s_i^0, n_i^0, s_0, and n_0) be non-negative integers (as in Martz *et al* (1988)). The demand availability posterior distributions for the subsystems and LPCI system, along with posterior quantiles, are displayed in Table C.1. Because there were no current binomial data available at the system and subsystem levels, the posterior distributions for the reliability these components are also the combined prior distributions for the reliability these components.

The system demand availability posterior mean is estimated to be 0.99999 with posterior standard deviation 1.09×10^{-5}. The posterior entropy is -10.56986. The ultimate goal of the initial study was to estimate the demand unavailability of the LPCI system. Martz and Waller (1990) note that the posterior distribution of demand unavailability for the LPCI system is Beta(0.78, 80745.70); the posterior mean demand unavailability is 9.6599×10^{-6}. Martz and Waller (1990) note that this can be interpreted as the LPCI system will be unavailable on demand, on average, in one out of every 103,000 demands. The standard deviation of the system demand unavailability posterior distribution is the same as that for the posterior distribution of system demand availability.

Analysis of First-Stage Data Using the Model of Johnson *et al* (2003)

The implementation of an analysis of these data using the model described in Section 1.2 is quite similar to the implementation described in Appendix B and is not repeated here. Trace plots (not included) indicate that a burn-in period of 25,000 is sufficient. Plots of the autocorrelation for the 26 model parameters are displayed in Figure C.1 and Figure C.2. Some model parameters showed non-negligible autocorrelation even after a lag of 500, thus the posterior draws were ultimately thinned to every 1000^{th} draw. The MCMC algorithm was run for 10,025,000 iterations and 10,000 posterior draws were collected.

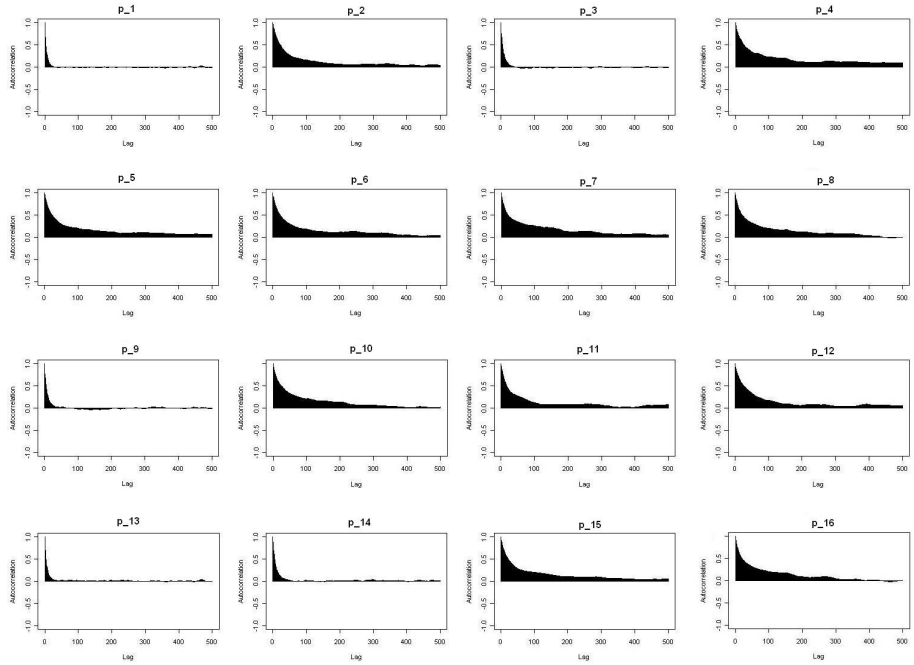


Figure C.1 Autocorrelation plots for Components 1 - 16

The estimated posterior means and standard deviations for the components' demand availability are displayed in Table C.2, along with naive and batch means Monte Carlo standard error estimates. The system demand availability posterior mean was estimated to be roughly 0.999997; this implies that an estimate of the mean demand unavailability is 3×10^{-6}. The standard deviation of the system demand availability (and demand unavailability) posterior distribution is estimated to be about 8.31×10^{-6}. The estimated posterior quantiles of the

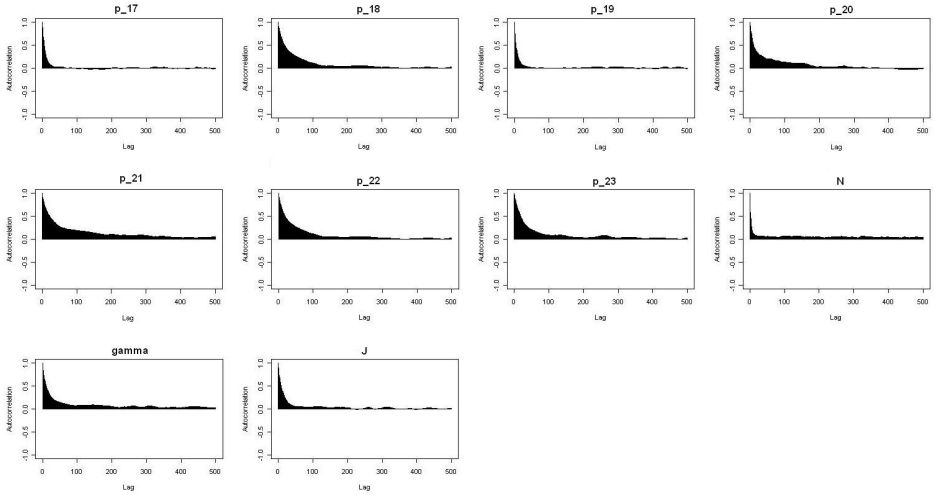


Figure C.2 Autocorrelation plots for Components 17 - 23, N, γ, and J

demand availability posterior distribution for each component are displayed in Table C.3.

Table C.2 Estimated posterior means and standard deviations, with naive and batch means Monte Carlo SE estimates, for the 26 model parameters

Parameter	Mean	Std. Dev.	Naive SE	Batch Means SE
p_1	0.983916128	0.007447976	0.000074476	7.76097E-05
p_2	0.999472288	0.001364488	1.36442E-05	1.39845E-05
p_3	0.991164781	0.005618576	5.61829E-05	5.33873E-05
p_4	0.999411519	0.001488701	1.48863E-05	1.59471E-05
p_5	0.999018263	0.001894815	1.89472E-05	0.000018406
p_6	0.999294016	0.001650847	1.65076E-05	1.77173E-05
p_7	0.998618999	0.002345578	2.34546E-05	2.29422E-05
p_8	0.999043649	0.001918492	0.000019184	2.04057E-05
p_9	0.991180182	0.005541155	5.54088E-05	5.06695E-05
p_{10}	0.999429241	0.001467134	1.46706E-05	1.62665E-05
p_{11}	0.999011489	0.001891313	1.89122E-05	1.79316E-05
p_{12}	0.99932327	0.001570879	0.000015708	1.58191E-05
p_{13}	0.983396692	0.007536764	7.53639E-05	7.75518E-05
p_{14}	0.990581316	0.005774819	5.77453E-05	5.33292E-05
p_{15}	0.998312896	0.002480738	2.48061E-05	2.56738E-05
p_{16}	0.997663559	0.002889602	2.88946E-05	2.82213E-05
p_{17}	0.99061443	0.005721233	5.72095E-05	5.40068E-05
p_{18}	0.998335358	0.00242819	2.42807E-05	2.41092E-05
p_{19}	0.999843829	0.000125813	1.2581E-06	1.1758E-06
p_{20}	0.999978136	3.43047E-05	0.000000343	3.325E-07
p_{21}	0.998156995	0.002486406	2.48628E-05	2.57664E-05
p_{22}	0.998313533	0.002429089	2.42897E-05	2.41183E-05
p_{23}	0.999996787	8.3091E-06	8.31E-08	8.78E-08
N	16.91521045	3.898667421	0.038984725	0.03861023
γ	0.973929975	0.013340609	0.000133399	0.000131045
J	6.116928085	2.433083117	0.024329615	0.023170917

Table C.3 Estimated posterior quantiles for the 26 model parameters

	Min	2.5%	5%	50%	95%	97.5%	Max
p_1	0.942891	0.966560	0.97023	0.984987	0.994014	0.995254	0.998736
p_2	0.982079	0.995437	0.996914	0.999986	1	1	1
p_3	0.953118	0.977228	0.980485	0.992315	0.998012	0.998567	0.999834
p_4	0.978847	0.994962	0.996635	0.999978	1	1	1
p_5	0.978192	0.993562	0.995319	0.999826	0.999999998	1	1
p_6	0.968391	0.994592	0.996352	0.999959	1	1	1
p_7	0.965134	0.991836	0.99393	0.999613	0.999999996	1	1
p_8	0.973222	0.993299	0.995152	0.999888	1	1	1
p_9	0.952095	0.977884	0.980744	0.992269	0.998012	0.998617	0.999886
p_{10}	0.977273	0.995164	0.996895	0.99998	1	1	1
p_{11}	0.965841	0.993511	0.995444	0.999809	0.999999998	1	1
p_{12}	0.979466	0.994869	0.996398	0.999958	1	1	1
p_{13}	0.94289	0.965891	0.969564	0.984471	0.993687	0.994874	0.998398
p_{14}	0.953118	0.976216	0.979554	0.991716	0.997717	0.998268	0.999834
p_{15}	0.968391	0.991148	0.993421	0.99927	0.999997	0.999999	1
p_{16}	0.965131	0.9898078	0.991925	0.998679	0.999977	0.999993	1
p_{17}	0.951286	0.976768	0.979742	0.991667	0.997756	0.998392	0.999886
p_{18}	0.965841	0.991151	0.993473	0.999302	0.999997	0.9999995	1
p_{19}	0.998539	0.999517	0.999603	0.999877	0.999974	0.999981	0.999998
p_{20}	0.999554	0.999886	0.999916	0.99999	0.9999999	0.99999995	1
p_{21}	0.968291	0.991013	0.993247	0.99911	0.999912	0.99994	0.999995
p_{22}	0.965808	0.991131	0.993459	0.99928	0.999986	0.999994	0.99999999
p_{23}	0.999795	0.999976	0.999985	0.9999995	0.999999994	0.999999998	1
N	4.183	10.091	11.079	16.608	23.805	25.584	36.666
γ	0.8666	0.9405	0.9487	0.9767	0.9895	0.991	0.9963
J	0.685	2.300	2.728	5.804	10.503	11.645	21.323

Table C.4 Sensitivity of system reliability posterior distribution to choice of prior distributions for N, J, and γ

Case	N	J	γ	μ	2.5%	50%	97.5%
1	Gam(5, 1)	Gam(5, 1)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.99999679	0.999976	0.9999995	0.999999998
2	Gam(5, 1)	Gam(5, 1)	Beta(1, 1)	0.99999655	0.99997380	0.999999383	0.9999999974
3	Gam(5, 1)	Gam(4, 1)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.99999682	0.99997505	0.999999511	0.9999999982
4	Gam(5, 1)	Gam(4, 1)	Beta(1, 1)	0.99999672	0.99997623	0.999999464	0.9999999979
5	Gam(4, 1)	Gam(5, 1)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.99999705	0.99997702	0.999999476	0.9999999982
6	Gam(4, 1)	Gam(5, 1)	Beta(1, 1)	0.99999670	0.99997565	0.999999478	0.9999999981
7	Gam(4, 1)	Gam(4, 1)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.99999702	0.99997619	0.999999568	0.9999999983
8	Gam(4, 1)	Gam(4, 1)	Beta(1, 1)	0.9999968	0.99997490	0.999999508	0.9999999982
9	Exp($\frac{1}{5}$)	Exp($\frac{1}{5}$)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.99999926	0.99999392	0.999999975	1
10	Exp($\frac{1}{5}$)	Exp($\frac{1}{5}$)	Beta(1, 1)	0.99999921	0.99999367	0.999999974	1
11	Exp($\frac{1}{5}$)	Gam(5, 1)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.99999856	0.99998764	0.999999904	1
12	Exp($\frac{1}{5}$)	Gam(5, 1)	Beta(1, 1)	0.99999848	0.99998767	0.999999893	1
13	Exp($\frac{1}{5}$)	Gam(4, 1)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.99999860	0.99998683	0.999999920	1
14	Exp($\frac{1}{5}$)	Gam(4, 1)	Beta(1, 1)	0.99999857	0.99998806	0.999999902	1
15	Gam(5, 1)	Exp($\frac{1}{5}$)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.99999801	0.99998377	0.999999765	0.9999999994
16	Gam(5, 1)	Exp($\frac{1}{5}$)	Beta(1, 1)	0.99999801	0.99998391	0.999999764	0.9999999994
17	Gam(4, 1)	Exp($\frac{1}{5}$)	Beta($\frac{1}{2}$, $\frac{1}{2}$)	0.99999813	0.99998494	0.999999795	0.9999999995
18	Gam(4, 1)	Exp($\frac{1}{5}$)	Beta(1, 1)	0.99999804	0.99998452	0.99999979	0.9999999995

Model Assessment and Sensitivity Analysis

We assess the model fit by using the Bayesian χ^2 goodness of fit test described in Appendix B. There are 12 observations available, thus the number of bins is approximately $K = 12^{0.4} \approx 3$. Roughly 8.4% of the $M = 10,000$ test statistics exceeded the 98% quantile of the χ^2_2 distribution, indicating no lack of fit.

To assess the sensitivity of the system demand availability posterior distribution to the choice of prior distribution for the parameters N, J, and γ, we consider 18 combinations of prior distributions for these parameters; the choice of the prior distributions selected for the sensitivity analysis is discussed in Appendix B. The system reliability posterior mean and 2.5%, 50%, and 97.5% posterior quantiles for the 18 combinations of prior distributions are displayed in Table C.4. In this example, changing the prior distribution for any of these parameters slightly does not seem to drastically impact the system reliability posterior distribution.

BIBLIOGRAPHY

Agresti, A. (2002) *Categorical Data Analysis, Second Edition.* Hoboken, NJ: John Wiley & Sons, Inc.

Anderson-Cook, C. M., Graves, T. L., Hamada, M., Hengartner, N., Johnson, V. E., Reese, C. S., and Wilson, A. G. (2007). Bayesian Stockpile Reliability Methodology for Complex Systems. *Journal of the Military Operations Research Society*, *12*(2), p. 25-37.

Anderson-Cook, C. M. (2009). Evaluating the Series or Parallel Structure Assumptions for System Reliability. *Quality Engineering* (in press).

Casella, G. and Berger, R. L. (2002) *Statistical Inference, Second Edition.* Pacific Grove, CA: Duxbury.

Chaloner, K. and Verdinelli, I. (1995). Bayesian Experimental Design: A Review. *Statistical Science*, *10*(3), p. 273-304.

Cover, T. M. and Thomas, J. A. (1991) *Elements in Information Theory.* New York: John Wiley & Sons, Inc.

De Jong, K. A. (2006). *Evolutionary Computation: A Unified Approach.* Cambridge, MA: The MIT Press.

Fitzpatrick, J. M. and Grefenstette, J. J. (1988). Genetic Algorithms in Noisy Environments. *Machine Learning*, *3*(2-3), p. 101-120.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004). *Bayesian Data Analysis, Second Edition.* New York: Chapman & Hall/CRC.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, Machine Learning*. Reading, MA: Addison-Wesley Publishing Company, Inc.

Graves, T. L (2001). "YADAS: An Object-Oriented Framework for Data Analysis Using Markov Chain Monte Carlo." Los Alamos National Laboratory Technical Report, LA-UR-01-4804.

Graves, T. L (2003). "An Introduction to YADAS." http://yadas.lanl.gov.

Graves, T. L. (2005). "Automatic Step Size Selection in Random Walk Metropolis Algorithms." Los Alamos National Laboratory Technical Report, LA-UR-05-2359.

Graves, T. L. and Hamada, M. S. (2004). "Combining Multi-Level Data to Assess System Reliability and Allocate Resources Optimally: Bayesian Methods and Computation." Los Alamos National Laboratory Technical Report, LA-UR-04-7157.

Graves, T. L., Hamada, M. S., Klamann, R. M., Koehler, A. C., and Martz, H. F. (2008). Using simultaneous higher-level and partial lower-level data in reliability assessments. *Reliability Engineering and System Safety*, *93*(8), p. 1273-1279.

Hamada, M. S., Martz, H. F., Reese, C. S., and Wilson, A. G. (2001). Finding Near-Optimal Bayesian Experimental Designs via Genetic Algorithms. *The American Statistician*, *55*(3), p. 175-181.

Hamada, M., Martz, H. F., Reese, C. S., Graves, T., Johnson, V., and Wilson, A. G. (2004). A fully Bayesian approach for combining multilevel failure information in fault tree quantification and optimal follow-on resource allocation. *Reliability Engineering and System Safety*, *86*(3), p. 297-305.

Hamada, M., Wilson, A. G., Reese, C. S., and Martz, H. F. (2008). *Bayesian Reliability*. New York: Springer.

Institute of Electrical and Electronic Engineers (IEEE) (1983). *IEEE Guide to the Collection and Presentation of Electrical, Electronic, Sensing Component, and Mechanical Equipment Reliability Data for Nuclear-Power Generating Stations*, New York: Author. (Distributed by Wiley-Interscience, New York.)

Johnson, V. E., Graves, T. L., Hamada, M.,and Reese C. S. (2003). "A Hierarchical Model for Estimating the Reliability of Complex Systems (with Discussion)," *Bayesian Statistics 7*, Oxford University Press.

Johnson, V. E. (2004). A Bayesian χ^2 Test for Goodness-of-Fit. *The Annals of Statistics*, *32*(6), p. 2361-2384.

Jones, G. L., Haran, M., Caffo, B. S., and Neath, R. (2006). Fixed-width output analysis for Markov chain Monte Carlo. *Journal of the American Statistical Association*, *101*(467), p. 1537-1547.

Lindley, D. V. (1956). On a Measure of Information Provided by an Experiment. *The Annals of Mathematical Statistics*, *27*(4), p. 986-1005.

Martz, H. F., Waller, R. A., and Fickas, E. T. (1988). Bayesian Reliability Analysis of Series Systems of Binomial Subsystems and Components. *Technometrics*, *30*(2), p. 143-154.

Martz, H. F. and Waller, R. A. (1990). Bayesian Reliability Analysis of Complex Series/Parallel Systems of Binomial Subsystems and Components. *Technometrics*, *32*(4), p. 407-416.

Michalewicz, Z. (1996) *Genetic Algorithms + Data Structures = Evolution Programs, 3rd Edition*. New York: Springer.

Plummer, M., Best, N., Cowles, K., and Vines, K. (2008). coda: Output analysis and diagnostics for MCMC. R package version 0.13-3.

Robert, C. P. and Casella, G. (2004). *Monte Carlo Statistical Methods, Second Edition*. New York: Springer.

Roberts, G. O. and Rosenthal, J. S. (2001). Optimal Scaling for various Metropolis-Hastings Algorithms. *Statistical Science*, *16*(3), p. 351-367.

Springer, M. D. (1979). *The Algebra of Random Variables*. New York: John Wiley.

Thompson, W. E. and Haynes, R. D. (1980). On the Reliability, Availability, and Bayes Confidence Intervals for Multicomponent Systems. *Naval Research Logistics Quarterly*, 27, p.345-358.

Weaver, B. P. and Hamada, M. S. (2008). A Bayesian Approach to the Analysis of Industrial Experiments: An Illustration with Binomial Count Data. *Quality Engineering*, *20*(3), p. 269-280.

Wilson, A. G., Graves, T. L., Hamada, M., and Reese C. S. (2006). Advances in Data Combination, Analysis and Collection for System Reliability Assessment. *Statistical Science*, *21*(4), p. 514-531.

Winkler, R. L. (1968). The Consensus of Subjective Probability Distributions. *Management Science*, *15*, p. B61-B75.